The LCD Animator 128

National Control Devices

128 Frames of Full-Screen Animation
64 Frames of Layered ASCII Text and Graphics
Supports 8 Frames of 16-Level Gray-Scale Imaging
High-Speed Animation up to 960 Frames per Second
Customized for our Backlit 256x128 Graphic LCD Display

Users Guide

Revision 1.02

By Ryan Sheldon

Quick-Start Begins on Page 1.

Copyright © 1998, 1999, National Control Devices

Forward:

The LCD Animator series are perhaps the most powerful and affordable graphic LCD display controllers available for RS-232 control applications. Programming the LCD Animator is reduced by the use of the LCD Animator Setup Utility. With this program, you will be able to use a simple graphical user-interface to configure the LCD Animator to your specific application. Nearly all of the functions that make the LCD Animator so powerful can be accessed by use of the LCD Animator Setup Utility. This provides a powerful interface for programmers and embedded systems designers by significantly speeding development.

This guide will provide users with a complete reference manual for hardware setup and programming of the LCD Animator. Start with the LCD Animator Setup Utility to program the display with the images and text that best suit your needs. Then, use your embedded system to send simple ASCII characters to page through your pre-loaded images. Complete control of the display is outlined in the programming section later in this guide. Here, you will find out how to send your own images and text to the display under your own program control.

There are 7 core subroutines responsible for all display functions. These routines are relatively short, usually 4 or 10 lines of actual code. The LCD Animator is probably the easiest display to program once the core subroutines have been ported. This guide will assist you in porting the code to your native language by providing a detailed explanation of what each line of code does.

We hope you find the LCD Animator to be as useful as it is powerful. In the future, we plan to provide additional setup utilities for the LCD Animator and the LCD Animator Lite demonstrating connection of both larger and smaller graphic displays. The LCD Animator hardware was designed to be practical and adaptable to future hardware and software needs. Visit our web site at http://members.aol.com/ncdcat/ for the latest software and developments. We have an ever-growing product line, and chances are good that you can find odd and unique computer-control products that will match your needs.

We also hope that users will provide feedback so that we can continue to focus our developmental efforts into applications that serve our customers well. Should you have any questions or comments regarding the use of the LCD Animator or any other NCD product, please contact Ryan Sheldon at (404) 244-2432.

Thanks:

I would like to thank the following people for their direct contribution to the LCD Animator project:

Tony Belehradek Jr., thanks for cutting the development time of this project in half.

Tom Maier, thanks for the PIC programming; this product would not exist without you.

Randy Schaffer, Earth Computer Technology, (714) 448-9368, thank you for the source of the LCD displays.

Jeff Sanchez, Northgate Manufacturing, (707) 786-9195, thanks for the assembly of the LCD Animator 128 boards. Jeff is just starting out and does EXCELLENT work; call him if you need assembly done in ANY volume.

Kima Dirks, thanks for all the editing; your patience and red marks are greatly appreciated and much needed.

Quickstart:

1) Connect the LCD Animator 128 to the LCD screen. Use the diagram shown on page 2; read the comments on page 3.
2) CONNECT 3-5 VOLT BATTERY TO B1+.
3) Connect a power supply to the LCD Animator 128. See the diagram on page 2. Make SURE the power is not connected backwards. DO NOT TURN ON THE POWER AT THIS TIME.
4) Connect the RS-232 data and RS-232 ground to the LCD Animator 128 as shown on page 2.
5) Connect the wires for the EL Backlight to the LCD Animator 128.
6) Verify jumpers are correctly set on the LCD Animator 128. Use the Default Jumper Settings Table on page 2.
7) Install the Setup Program for Windows 95. Start with floppy disk 1; leave disk 2 in drive for the QBasic tutorial.
8) SHOCK HAZARD: Apply power to the LCD Animator. DO NOT TOUCH THE EL BACKLIGHT WIRES OR CONNECTORS WITH POWER APPLIED.
9) Run the LCD Animator Setup Utility that you just installed.
10) Click on the “SEND IMAGE” button.
11) Contrast settings are adjusted prior to shipment; adjust as necessary. See page 2 for the location of the contrast adjustment knob.

You MUST attach batteries to the Battery Backup Connectors for Reliable Bootup of the LCD Animator 128 Logic Board.

WARNING: The EL Inverter Produces High Voltages on Powerup. Never Touch the Connectors of the LCD Animator or the LCD Display While in Operation.

SLEEP: The LCD Animator can be put to sleep by attaching a momentary button (normally open) between the G and S pins on the SLEEP header. When the button is pushed, the LCD Animator is put to sleep and will not wake up until power is reapplied to the LCD Animator board.

If you choose to attach external logic circuits to the G (ground) and S pins, make sure a high-impedence circuit is used. The S pin is interfaced with the BAUD SELECT system, and Baud-Rate configuration may be undetectable if an incompatible circuit is used. For this reason, the G and S pins must NEVER be connected when the LCD Animator is first powered up. Doing so may damage the custom network interface processor.

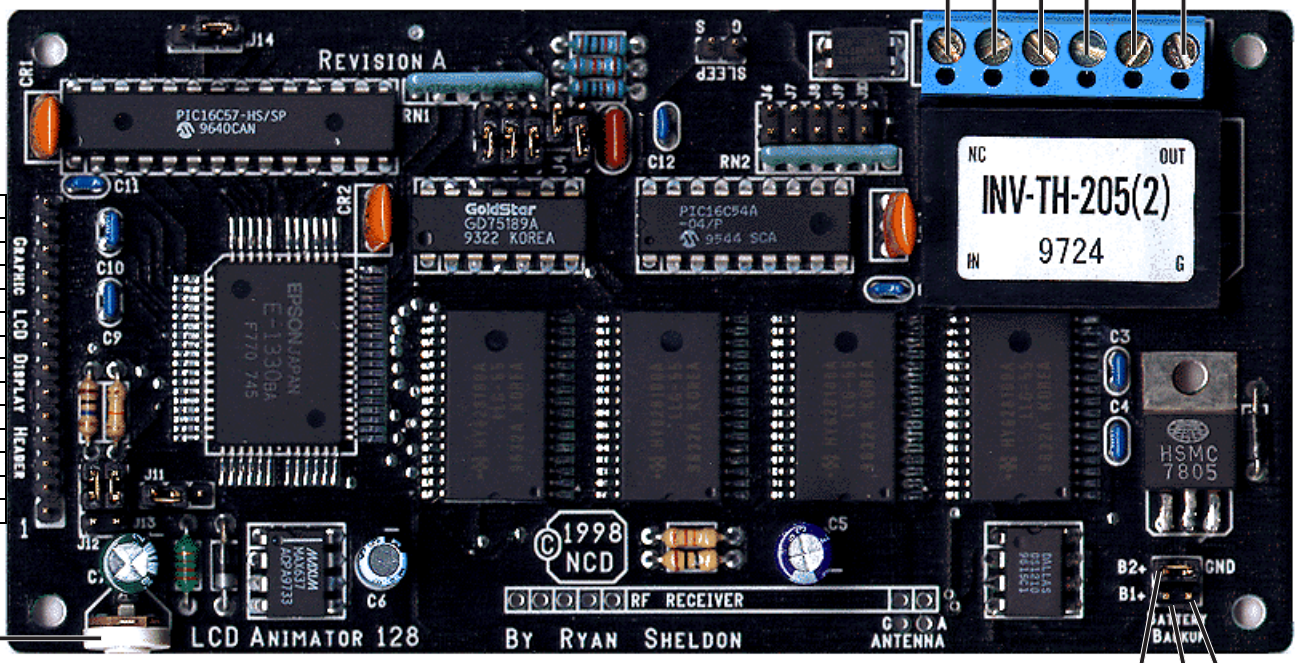
BACKLIGHT WIRING NOTE: The LCD Display has 2 wires connected to the right side. These 2 wires are used to supply power to the backlight and are NOT polarity sensitive. Connect to EL Ground and Power Terminals in any order.

EL Ground
EL Power (WARNING)
RS-232/Logic Ground
RS-232 Data Input
+5 Input (Used for Backlight)
+12 Input (Used for Logic)

Note: Pin 1 or the EPSON Display Header Connects to Pin 1 or LCD Display.

EPSON Display Header

D3	14
D2	13
D1	12
D0	11
XECL	10
XSCL	9
DIN	8
YSCL	7
YDIS	6
FR	5
LP	4
VLCD	3
GND	2
V+	1



Contrast Adjust

RF Receiver Option - See Docs for Receiver.

Battery 2 V+ 2-5 volts
Battery 1 V+ 2-5 volts
Battery Backup Ground

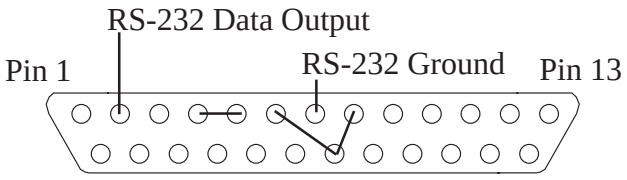
Jumper Description and Factory Default Configuration

Jumper	Function	Install Notes	Default Setting
J1	Animator Address 1	-	Not Installed
J2	Animator Address 2	-	Not Installed
J3	Animator Address 3	-	Not Installed
J4	Animator Baud 1	-	Installed
J5	Animator Baud 2	-	Not Installed
J6-J10	Reserved	Reserved	Never Installed
J11	Display Type	128 Frames of Graphics (Install Left) 64 Frames of Graphics/Text (Install Right)	Install Between Right Terminals
J12	Display Compatibility	High-Impedance Interface (Install Up) Direct Drive Interface (Install Down)	Install Between Upper Terminals
J13	Display Compatibility	High-Impedance Interface (Install Up) Direct Drive Interface (Install Down)	Install Between Upper Terminals
J14	Animator RS-232 Select	Data From RF Receiver (Install Left) Data from RS-232 (Install Right)	Install Between Right Terminals

NOTE: Up to 2 Batteries can be Connected to the LCD Animator for Memory Backup Protection. If a second battery is NOT installed, install a jumper between the B2 and GND connectors. Never Connect Batteries Backwards and Never Exceed 5 volts.

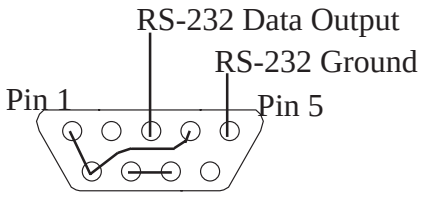
LCD Animator RS-232 Connections

25-Pin Serial Interface



Example Shown Using a DB-25 Female Connector

9-Pin Serial Interface



Example Shown Using a DB-9 Female Connector

The diagrams at left show how easy it is to connect the LCD Animator to the RS-232 port of your computer. The RS-232 Ground must share the ground of the LCD Animator. The RS-232 Data Output must be connected to the RS-232 Data Input on the LCD Animator.

The diagrams also show connections between pins on the RS-232 connector that are not connected to the LCD Animator. These connections disable hardware handshaking. Since handshaking is disabled in software, these connections may be omitted.

Note that both the DB-25 and DB-9 are female connectors. These connectors are readily available at your local Radio Shack as well as DigiKey, Mouser, Jameco, and many other places.

About LCD Animator Addressing:

The LCD Animator has 3 on-board jumpers that are used for setting the device number. In software, you will be able to speak to several LCD Animators attached to a single serial port by simply changing the device number to match the jumper settings of the device you want to control.

For example, let's say you have 2 LCD Animators attached to a single serial port and you want to turn on the backlight of each one. Set the address of the first LCD Animator to NCD Address 3 (see table at right). Set the device number of the second LCD Animator to 5. To turn on the backlight of both displays, make the following subroutine calls:

```
LIGHT 3, 1 : Rem Turn On the Backlight of Device 3.
LIGHT 5, 1 : Rem Turn On the Backlight of Device 5.
LIGHT 3, 0 : Rem Turn Off the Backlight of Device 3.
LIGHT 5, 0 : Rem Turn Off the Backlight of Device 5.
```

The LCD Animator counts as 2 devices on the NCD RS-232 control network. When set to address 3, the LCD Animator also listens to data at address 11. Address 3 is used for communicating data to the controller, while address 11 is used for communicating commands. NEVER USE A DEVICE NUMBER ABOVE 7 IN SOFTWARE. The provided routines automatically add 8 to the device number for sending commands.

LCD Animator Address Settings

NCD Address	J1	J2	J3
0,8	Removed	Removed	Removed
1,9	Installed	Removed	Removed
2,10	Removed	Installed	Removed
3,11	Installed	Installed	Removed
4,12	Removed	Removed	Installed
5,13	Installed	Removed	Installed
6,14	Removed	Installed	Installed
7,15	Installed	Installed	Installed

NOTE: Jumper settings are read only when power is first applied to the board. Changing jumpers while in operation will have no effect on most settings until the next time power is applied.

Baud Rate Selection

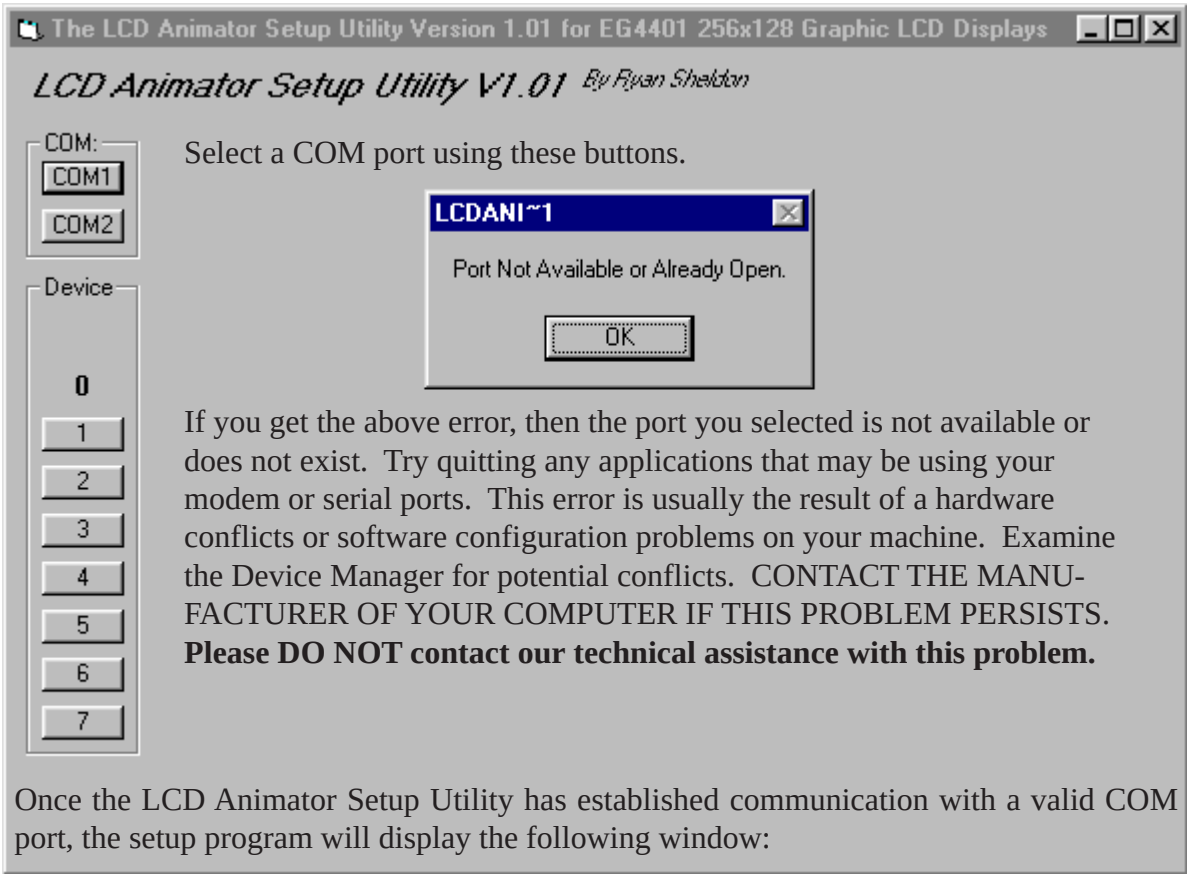
Baud Rate	J4	J5
9600	Removed	Removed
19,200	Installed	Removed
38,400	Removed	Installed
115,200	Installed	Installed

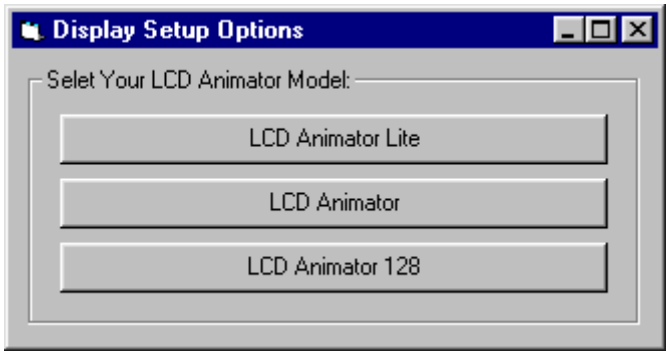
Connecting a Graphic Display to the LCD Animator:

At the time of writing, we are supporting only the 256x128 graphic display. This display is manufactured by EPSON and connects to the EPSON header of the board. Note that PIN 1 on the EPSON display MUST MATCH PIN 1 ON THE LCD ANIMATOR BOARD. Damage may result to the LCD Animator or the EPSON display if connected backwards. Pin 1 is marked on the LCD display. When the screen is facing down and the 14-pin connector is at the bottom, pin 1 is located on the left-most side.

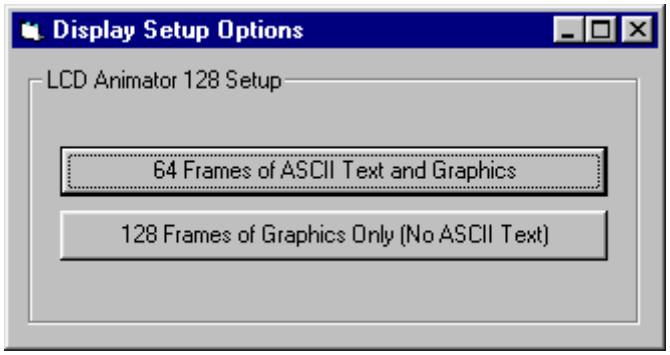
Please visit us on-line for the latest display connection information and software:

http://members.aol.com/ncdcat/


The LCD Animator Setup Utility Version 1.01 for EG4401 256x128 Graphic LCD Displays
LCD Animator Setup Utility V1.01
By Ryan Sheldon
COM:
COM1
COM2
Device
0
1
2
3
4
5
6
7
Select a COM port using these buttons.
LCDANI~1
Port Not Available or Already Open.
OK
If you get the above error, then the port you selected is not available or does not exist. Try quitting any applications that may be using your modem or serial ports. This error is usually the result of a hardware conflicts or software configuration problems on your machine. Examine the Device Manager for potential conflicts. CONTACT THE MANUFACTURER OF YOUR COMPUTER IF THIS PROBLEM PERSISTS.
Please DO NOT contact our technical assistance with this problem.
Once the LCD Animator Setup Utility has established communication with a valid COM port, the setup program will display the following window:



Next, select the LCD Animator 128 to customize communication to the hardware. Note that this window is displayed only if a valid serial port can be accessed.


Display Setup Options
LCD Animator 128 Setup
64 Frames of ASCII Text and Graphics
128 Frames of Graphics Only (No ASCII Text)

The next option will configure the LCD Animator 128 for your specific application. If your application requires you to send ASCII text to the display, select the top button. If your application does not require ASCII text, select the bottom button. This will configure the LCD Animator to use text memory for graphics, effectively doubling the image capacity.

COM: Active COM port.

Net A-D: Requires the NCD-QNET adapter, allowing up to 32 LCD Animators to be controlled from a single serial port. Up to 8 LCD Animators can be controlled per network. This button provides access to 4 networks, and has no effect without the QNET adapter.

Device: Use these buttons to configure multiple LCD Animators attached to the same serial port. Buttons can be changed at any time. Note that each LCD Animator must be set to a different device number in hardware.

Go to Sleep: Press this button to put the LCD Animator in SLEEP mode. Power should be removed from the LCD Animator when in sleep mode. Batteries MUST be attached to the LCD Animator's battery backup system for proper operation.

Backlight: By default, the backlight is turned on and an X will appear in this box. Click here to toggle the status of the backlight.

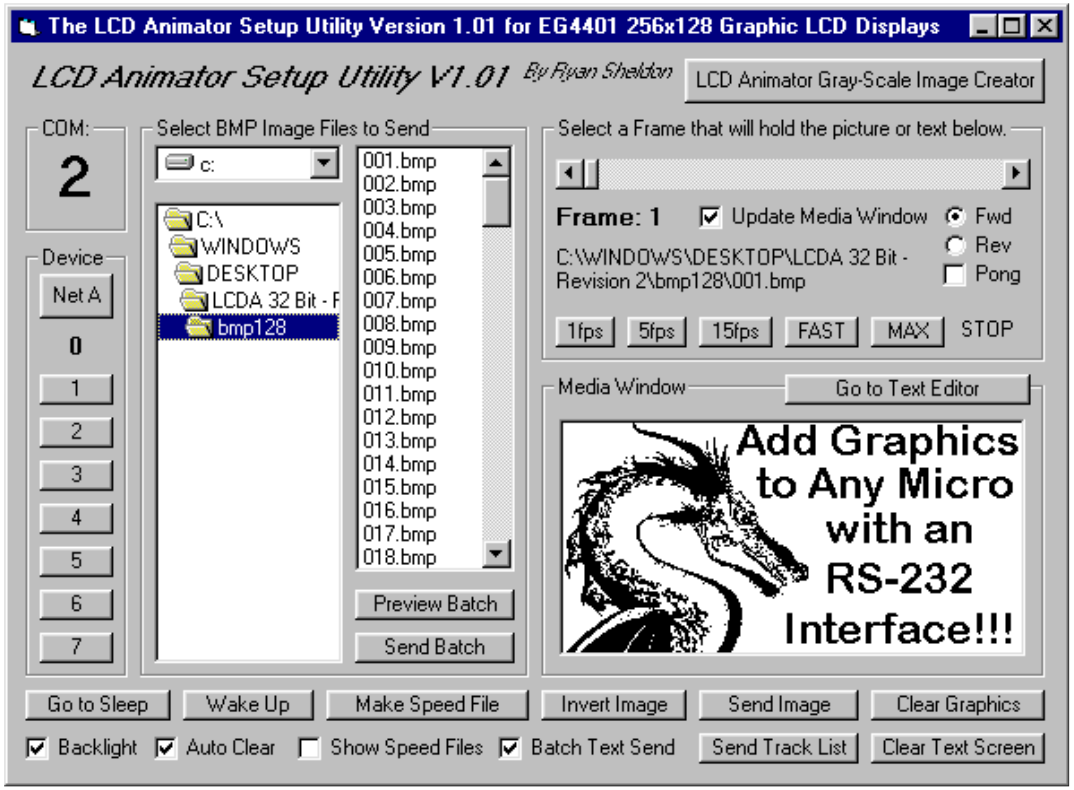
The LCD Animator Setup Utility provides important configuration information for the LCD Animator. Because of this, the setup utility MUST be run to properly initialize the LCD Animator and MUST be used if the battery backup system is ever accidentally disconnected.

Configuration information stays in the LCD Animator for the life of the batteries attached to the battery backup system. The LCD Animator should retain its memory for an estimated 5 years using 2 AA batteries, provided the LCD Animator is placed in sleep mode BEFORE power is removed.

Select Drive and Directory: Select an active location of BMP image files and/or SPD files.

Select File: Select a file to be previewed in the Media Window. Note that only .BMP (bitmap images) and .SPD (speed files) can be displayed.

Frame Slider: Move the slider to access any frame on the LCD Animator. As the slider is moved, the Media Window will change.



Gray-Scale Image Creator: Creates and displays 16-level gray-scale images for use with the LCD Animator 128 only.

Animation Controls: The Animation Controls select how fast the images are changed on the display in approximate frames per second (fps). Press STOP at any time to halt animation at 1, 5, or 15 fps. The FAST and MAX buttons are momentary. Note that the Media Window is updated at 1, 5, and 15 fps if the Update Media Window check is visible. FWD and REV control the direction in which the animation is played. The PONG button plays the animation forward, then backward, in a continuous loop.

Go to Text Editor: Replaces the graphic window with a text window that will send text to the display when the window is double-clicked.

Media Window: Allows the user to preview images BEFORE sending them to the LCD Animator.

Clear Graphic Screen: Clears the LCD's graphic screen on the currently selected frame. Does not erase any text.

Clear Text Screen: Clears the LCD's text screen on the currently selected frame. Does not erase any graphics.

Wake Up: This command wakes the LCD Animator from sleep mode. For proper operation, the battery backup system must have batteries attached and power must be applied to the logic circuits of the controller.

Autoclear: When enabled, the LCD Animator Setup Utility will automatically clear the text screen when new images are being uploaded to the display. This function is useful when initially powering the display and garbage is displayed on the text screen.

Preview Batch Files: Uses the Media Window to show animation.

Send Batch: Uploads up to 128 images to the display into each of the 128 frames. Note that BMP image files must be named from 001.BMP to 128.BMP. Any missing files will be skipped.

Cancel Batch: Button appears ONLY during batch send.

Make Speed File: Speed files are a raw dump of a bitmap image into a BYTE format. A speed file can be easily read by ANY programming language, allowing any micro or computer to send images to the display.

Show Speed Files: Shows the speed files in the file list.

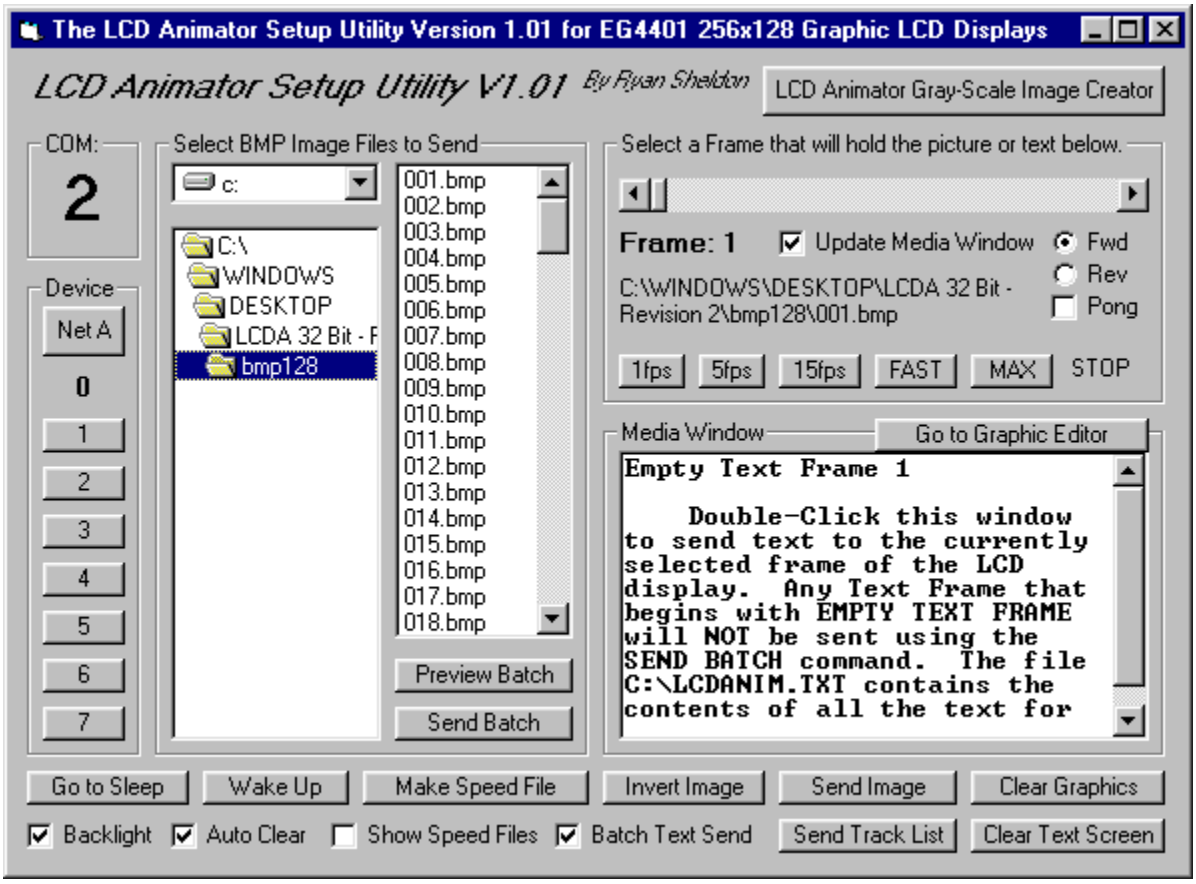
Invert Image: Inverts the image in the Media Window.

Batch Text Send: Sends ASCII text stored in C:\LCDANIM.TXT along with batch graphics.

Go to Text Edit Mode: Replaces the Graphic Box with a text editor that can be used for directly sending text to the display without any programming.

Send Image: Sends the image displayed in the Media Window to the currently selected frame of the LCD Animator.

Send Track List: Sends all available image files listed in the text file C:\TRACKLST.TXT.



Notes:

When the LCD Animator Setup Utility is closed, the display is put to sleep, and track lists and text entries are saved. In addition, the LCD Animator is put to sleep via the software sleep command. The software sleep mode can be recovered by restarting the LCD Animator Setup Utility or by issuing the Wake-Up command. When the hardware sleep mode is issued, the LCD Animator MUST be powered down.

Speed Files and the File List:

The file list can show both SPD (speed) and BMP (bitmap) image files. Speed files can be hidden by turning off the "Show Speed Files" option. A scroll bar will appear if the list becomes longer than the length of the window, as shown above. Speed files can be sent to the LCD Animator Setup Utility but will be displayed only in the Media Window if the original BMP image is available.

Speed files are created by first loading the BMP image into the Media Window. The program literally reads the pixels off the screen to determine what goes in the file.

The program starts at the upper left corner of the picture. It reads the on/off status of the first 8 pixels sitting next to each other to form a byte of data. The value of this byte is always 0 to 255 (because there are only 8 bits), and this value is written to the speed file. The program then looks at the next set of 8 pixels and stores the equivalent byte. This process continues in a right-reading, top-down order until the end of the image is reached. A 256x128 graphic display requires 4096 bytes to fill the display. Subsequently, there are 4096 data entries into the speed file.

Text:

The LCD Animator Setup Utility provides a simple method of sending text to the LCD display. By clicking the "Go to Text Editor" button, a window, will appear that acts as a word processor. Type your text in this window and double-click the window to send your text to the display. Sometimes, spaces will need to be added to get the text to align properly. You should avoid using the return key for a more reliable representation of what text will look like on the display.

Every text entry is recorded in the file C:\LCDANIM.TXT. Up to 64 text entries can be found in this file, depending on the LCD Animator model you are using. Note that ANY text entry that begins with "Empty Text Frame" will NOT be sent to the display using the Send Batch function.

Text and Graphic Layering:

Text is displayed in a separate layer of memory, allowing text to be altered without affecting the graphics. In addition, graphics may be changed without altering the text. Keep in mind that every frame consists of a text and graphic layer, but only when the text mode is enabled. When the text mode is disabled, the LCD Animator can use the memory reserved for text as graphic memory, permitting control of 128 different frames of graphics.

Graphic and Graphic/Text modes are configured in software as well as hardware. Jumper J11 configures the LCD Animator for these display modes. This jumper can be changed while the LCD Animator is powered. Please see the hardware diagram and associated descriptions on page 2 for details.

Also note that when the "Autoclear" check is visible, the text layer is automatically cleared during an image upload or batch upload. This feature is very useful for clearing garbage characters out of memory when using the LCD Animator for the first time, but may be undesirable after initial setup.

Gray-Scale Imaging:

Gray-Scale imaging allows you to add more complexity to your embedded system. Detailed images can be displayed on ANY monochrome LCD screen with the LCD Animator.

The LCD Animator is capable of displaying up to 8 16-level gray-scale images or 16 8-level gray-scale images. Gray-scale imaging is achieved by converting a 16-level gray-scale image into a set of 16 frames that can be animated at high speed. When animated, the display produces the illusion of a 16-level gray-scale image. Because LCD displays are slow to respond to frame changes, image flicker is virtually nonexistent, resulting in the display of a rock-solid 16-level gray-scale image on a traditionally monochrome LCD display.

Each level of a gray-scale image occupies a single frame of memory. For best results, 19.2K baud or higher should be used for animation. If you purchased the gray-scale image converter utility, the "LCD Animator Gray-Scale Image Creator" button will be available for use in the top right corner of the "LCD Animator Setup Utility" screen. This utility converts your images into a set of frames and allows you to preview your images. In addition, you can eliminate frames that have no visual effect on the gray-scale image while the image is displayed on the LCD screen. In most applications, only 10-14 frames are needed to accurately represent a gray-scale image. Typically, the first and last frames can be eliminated without ANY visual loss.

Automated LCD Animator Setup:

The LCD Animator Setup Utility provides many functions that will greatly speed upload time. Because LCD Animator 128 users will not want to upload all 128 images manually, we employ 3 types of batch transfers designed to minimize the number of clicks required to upload all 128 frames of graphics or 64 frames of graphics and text (yes, this program employs batch TEXT uploads to all 64 frames as well).

Send Batch Image Transfers:

There are 2 types of batch image transfers available to the user. The Send Batch button provides a very simple and powerful way to upload several images to the display unattended. The Send Batch function works only with image frames numbered 001.BMP to 128.BMP. Note that this naming system is REQUIRED for the Send Batch function to work. Once clicked, the Send Batch function looks for frames with this numbering system in the currently selected file list. If found, the image will be uploaded to the corresponding frame. If missing, the file and frame will be skipped.

Batch Preview:

The Batch Preview button allows you to see the sequence of images that will be sent to the display in the Media Window of the LCD Animator Setup Utility software. Batch transfers may be canceled by selecting the Cancel Batch button, which replaces the Send Batch button during batch transfers.

Batch Upload Tutorial:

To experiment with batch uploads, first find the BMP128 directory created when you installed the LCD Animator Setup Utility software. Second, unzip the BMP128.ZIP file to reveal the 128-frame animation. Third, locate the BMP128 directory in the LCD Animator Setup Utility software. A scrollable list of image files will be displayed. Finally, select the Send Batch button to upload all 128 images into the LCD Animator display ram.

It is also possible to upload a specific set of files to the display. For example, to upload frames 32, 63, and 120 ONLY into the display, copy the corresponding image files 032.BMP, 063.BMP, and 120.BMP into an empty directory. Next, select the directory in the file list that holds these three images. Select the Send Batch button to transfer these images into frames 32, 63, and 120. All other frames will be skipped.

Track List Batch Image Transfers:

The second method of uploading a batch of images to the display makes use of a track list. This file, located in C:\TRACKLST.TXT, contains a list of images to be sent to the LCD Animator. Every image you send to the LCD Animator is recorded into this track list, which is a simple list with 128 entries containing the full path and filename of each frame. Please note that the Send Batch button also modifies items in the track list.

Track lists are useful if you are frequently changing and uploading images located in different areas of your hard drive. To send every file recorded in the track list, select the "Send Track List" button. Missing files will be skipped, along with the corresponding frame.

Batch Text Transfers:

Graphics and text frames are treated as equals under the LCD Animator Setup Utility. For every frame of graphics, a frame of text may be overlaid. The LCD Animator 128 supports a maximum of 64 frames of graphics overlaid with text or 128 frames of graphics with no text capabilities. The text layer can be manipulated without altering the graphic layer. Similarly, graphics can be changed behind the text. Just as image file references are stored in the track list, C:\LCDANIM.TXT is dedicated to holding 64 frames of text with a single frame holding a maximum of 512 characters of text.

When the Batch Text Send check is visible, all 64 frames of text will be sent to all 64 frames of the LCD Animator. Any frames of text that begin with "Empty Text Frame" will NOT be sent to the display. To send a batch of text to the display, select the Send Batch button. Both images and text will be transferred to the display. To send text only, locate an empty directory in the file list of the LCD Animator Setup Utility and select the Send Batch button. Since no images are available for transfer to the display, all images will be skipped, but all 64 frames of text will be transferred.

Programming:

The LCD Animator is capable of performing all operations with the help of only a few subroutines. If you are intimidated about porting a bunch of code to the LCD Animator, let me end your worries here. There are only about 7 routines that need to be ported, they are all very short (usually around 6 lines of actual code), and they are all easy to translate to any programming language.

Note: Before proceeding, connect the LCD Animator to your computer and run the "LCD Animator Setup Utility." This will configure the LCD Animator for the LCD display, significantly reducing the amount of code required to communicate to the LCD Animator. Also note that the backup batteries MUST BE ATTACHED. See the LCD Animator Connections Guide for details.

Addressing:

As you know by now, the LCD Animator conforms to the NCD RS-232 control network protocol. This protocol allows up to 16 NCD devices to share a single serial port in any combination. The LCD Animator counts as 2 devices on the NCD RS-232 control network.

A set of jumpers on the board will determine the LCD Animator's location on the network. Typically, device numbers range from 0 to 15. Since the LCD Animator counts as 2 devices, the device number ranges from 0 to 7. So when you call any of the routines below, you must specify a device number that ranges from 0 to 7.

When the LCD Animator is set to address 0, it is also listening to address 8. Similarly, when the LCD Animator is set to address 1, it is also listening to address 9. Adding 8 to the device number determines the second address the LCD Animator is listening to.

The following section will help you understand how the addressing system works. If you are not interested in the technicalities of this protocol, you can skip this section entirely and just port the example code. This section is provided for educational purposes only, and your understanding of it is not required to port the code to your native language.

Protocol:

Formatting data for communication to the LCD Animator is achieved by using one of 2 equations:

DEV=DEVICE*16	Used to send data to the LCD Animator.
DEV=(DEVICE+8)*16	Used to send commands to the LCD Animator.

The following demonstrates sending a command to the LCD Animator using the second of the 2 equations. To turn the backlight on or off, the user needs to provide a device number and a status variable. Valid device numbers range from 0 to 7, and a valid status variable can equal 1 or 0.

Examples:

LIGHT 0,1	Call Light Sub, Talk to Device 0, Turn Light On (1).
LIGHT 5,1	Call Light Sub, Talk to Device 5, Turn Light Off (0).

When the LIGHT subroutine is called, the first thing it does is calculate which LCD Animator to talk to. To do this, it takes the device number you fed the routine (from 0 to 7) and adds 8 to it. Adding 8 tells the LCD Animator to speak to the upper set of addresses (0-7 are the lower set for sending data to the LCD Animator, and 8-15 are the upper set for sending commands to the LCD Animator).

```
Procedure LIGHT[DEVICE,STATUS]
   'Compute Device Number
   DEV=(DEVICE+8)*16
   If STATUS=1
      'Send the Command to Turn On Backlight
      Serial Send 0,Chr$(DEV+1)
   Else
      'Send the Command to Turn Off Backlight
      Serial Send 0,Chr$(DEV+0)
   End If
End Proc
```

Next, the LIGHT routine determines whether the light should be on or off. If the STATUS equals 1, send command 3. If the user wants to turn off the light, send command 2.. To send this command to the correct device, the computed device number must be added to the command number.

ASCII 130 turns off the light off an LCD Animator set to Address 0.
ASCII 131 turns on the light on an LCD Animator set to Address 0.
ASCII 146 turns off the light on an LCD Animator set to Address 1.
ASCII 147 turns on the light on an LCD Animator set to Address 1.
ASCII 162 turns off the light on an LCD Animator set to Address 2.
ASCII 163 turns on the light on an LCD Animator set to Address 2.
ASCII 178 turns off the light on an LCD Animator set to Address 3.
ASCII 179 turns on the light on an LCD Animator set to Address 3.
ASCII 194 turns off the light on an LCD Animator set to Address 4.
ASCII 195 turns on the light on an LCD Animator set to Address 4.
ASCII 210 turns off the light on an LCD Animator set to Address 5.
ASCII 211 turns on the light on an LCD Animator set to Address 5.
ASCII 226 turns off the light on an LCD Animator set to Address 6.
ASCII 227 turns on the light on an LCD Animator set to Address 6.
ASCII 242 turns off the light on an LCD Animator set to Address 7.
ASCII 243 turns on the light on an LCD Animator set to Address 7.

The table at left simplifies the addressing system for controlling the backlight on 8 LCD Animators attached to a single RS-232 serial port.

It is also important to note the slight difference between the device number and the address with regard to the LCD Animator. On most NCD devices, the device number and the address are the same thing. But because the LCD Animator counts as 2 devices on the network, it has 2 possible addresses for every device number.

All NCD devices are designed to respond to their address determined by their device number. The device number is set using 3 jumpers on the LCD Animator, labeling the device from 0 to 7 and determining which addresses the LCD Animator will listen to.

Device	Communicates with the Data Lines of the LCD Animator.
0	ASCII characters 0-15 are seen only by address 0.
1	ASCII characters 16-31 are seen only by address 1.
2	ASCII characters 32-47 are seen only by address 2.
3	ASCII characters 48-63 are seen only by address 3.
4	ASCII characters 64-79 are seen only by address 4.
5	ASCII characters 80-95 are seen only by address 5.
6	ASCII characters 96-111 are seen only by address 6.
7	ASCII characters 112-127 are seen only by address 7.
	Communicates with the Command Lines of the LCD Animator.
0+8	ASCII characters 128-143 are seen only by address 8.
1+8	ASCII characters 144-159 are seen only by address 9.
2+8	ASCII characters 160-175 are seen only by address 10.
3+8	ASCII characters 176-191 are seen only by address 11.
4+8	ASCII characters 192-207 are seen only by address 12.
5+8	ASCII characters 208-223 are seen only by address 13.
6+8	ASCII characters 224-239 are seen only by address 14.
7+8	ASCII characters 240-255 are seen only by address 15.

Understanding the Table:

A total of 256 different ASCII characters can be sent out the serial port of any computer or microcontroller. ASCII characters 0 to 127 are responsible for sending data (such as text and graphics) to LCD Animators set to devices 0 to 7. ASCII characters 128 to 255 are responsible for sending control commands (such as positioning the cursor or turning on/off the backlight) to LCD Animators 0 to 7.

Eight is added to the device number in hardware and in software. The control routines will route all data to NCD addresses 0 to 7 and all control information to NCD addresses 8 to 15.

So, speaking to an LCD Animator set to address 0 will consist of sending all data to the display using ASCII characters 0 to 15, and sending all control commands to the display will be accomplished using only ASCII characters 128 to 143.

End of Protocol: The rest is required reading for programming the LCD Animator.

Porting the Code:

As promised, porting the code to the LCD Animator will prove to be a painless operation. There are a few core routines that MUST be ported; other routines are optional. The core routines speak to the LCD Animator hardware directly, sending either commands or data to the display.

As discussed in the protocol section, these routines will handle all device addressing for you. So translating these routines with proper programming structure will enable you to later add more LCD Animators or other NCD devices on a single serial port.

Core Routines:

You should never directly transmit an ASCII character to the LCD Animator without using one of the provided core routines. The core routines were designed to be network compliant and will eliminate programming errors associated with sending data or commands to the incorrect device.

The following are all core routines that can be called to perform ANY operation the LCD Animator has to offer. The core routines have been optimized to be modular and very small. Using these core routines will significantly reduce the amount of programming required to speak to the LCD Animator while maintaining proper programming structure.

The core routines were translated from Amos Professional on the Amiga platform to Visual Basic 4 on the PC platform. Translation took exactly 15 minutes. Troubleshooting took approximately 1 hour. The display would not respond when the code was executed. The problem was located in the CD routine below. The CDS variable was not properly globalized and the if/then conditions for the CDS variable were mistyped.

```
Sub CD[Device,Status]
       'Compute Device Number
       DEV=(Device+8)*16
       ‘Determine if Status Line is Requested On
       If Status=1 then
              ‘If the CD line is NOT already on
              If CDS<>1 then
                     ‘Remember that it is on now
                     CDS=1
                     'Turn On CD Line
                     Serial Send Chr$(DEV+3)
              End If
       ‘Otherwise, turn off the status line
       Else
              ‘If the CD line is NOT already on
              If CDS<>0
                     ‘Remember that it is off now
                     CDS=0
                     'Turn Off CD Line
                     Serial Send Chr$(DEV+2)
              End If
       End If
End Sub
```

CD (Device, Status)

Command/Data Select. This subroutine is responsible for putting the LCD Animator in command or data mode. This routine is never called by the user and is required by another subroutine. The CDS variable increases the speed of image transfers by about 50%. Every time this subroutine is called, the value of the CDS variable is read and changed only when necessary. Make sure the CDS variable is globalized in your program. When the routine is exited, the stored value will need to be recalled by this subroutine.

```
‘This simplified version of the CD subroutine
‘may be used, but is much less efficient.
Sub CD[Device,Status]
   'Compute Device Number
   DEV=(Device+8)*16
   ‘Determine if Status Line is Requested On
   If Status=1 then
      'Turn On CD Line
      Serial Send Chr$(DEV+3)
   ‘Otherwise, turn off the status line
   Else
      'Turn Off CD Line
      Serial Send Chr$(DEV+2)
   End If
End Sub
```

```
Sub SENDBYTE [Device,Byte]
       'Compute Device Number
       DEV=Device*16
       'Compute Least Significant Nibble
       LSN=(Byte and 15)
       'Compute Most Significant Nibble
       MSN=(Byte and 240)/16
       'Send Least Significant Nibble
       Serial Send Chr$(LSN+DEV)
       'Send Most Significant Nibble
       Serial Send Chr$(MSN+DEV)
End Sub
```

SENDBYTE (Device, Byte)

This subroutine is responsible for sending a byte of data to the LCD Animator. It is required by other subroutines and is not used by the programmer. When the LCD Animator is in command mode, the byte sent by this routine is interpreted as a command. When in data mode, the byte sent is either an ASCII character or a graphics element.

This routine takes a byte and splits it into 2 nibbles. These nibbles are labeled LSN and MSN for least and most significant nibble. Each nibble is then sent to the LCD Animator at the correct address (DEV).

LCD Animator Control Routines (continued)

```
Sub FRAME [Device, F]
        'Compute the device number
        DEV=(Device+8)*16
        'FR can equal 0 to 15 ONLY.
        fr = f - 1: Chapter = 1
        While fr > 15
                Chapter = Chapter + 1
                fr = fr - 16
        Wend
        'Send Page Change Command
        Serial Send 0,Chr$(DEV+4)
        'Send Frame to Change To
        Serial Send 0,Chr$(DEV+fr)
        'Put a wait statement here of
        'about 7 to 14 µS.
        'This wait statement is important for
        'proper operation at 115.2K baud.
        'Send the Chapter Change Command
        'Chapter = 1 to 8
        Serial Send 0, Chr$(Dev + 5 + Chapter)
End Sub
```

```
Sub LIGHT [Device, Status]
        'Compute Device Number
        DEV=(Device+8)*16
        If Status=1
                'Turn On Backlight
                Serial Send 0,Chr$(DEV+1)
        Else
                'Turn Off Backlight
                Serial Send 0,Chr$(DEV+0)
        End If
End Sub
```

```
Sub CMD [Device, Byte]
        CD Device,1 : Rem Command Mode
        SENDBYTE Device, Byte
End Sub
```

```
Sub DAT [Device, Byte]
        CD Device,0 : Rem Data Mode
        SENDBYTE Device, Byte
End Sub
```

FRAME (Device, Frame)

The FRAME subroutine is responsible for changing frames on the LCD Animator. There are 128 possible values for **F** (frame) ranging from 1 to 128 when in the Graphic-Only mode. Use values of 1 to 64 when in the graphic/text mode.

LIGHT (Device, Status)

The LIGHT subroutine is responsible for turning the backlight of the display on or off. This subroutine is called by the user and is useful only if you have a backlit display attached to the LCD Animator.

CMD (Device, Status)

The CMD subroutine is required for sending a command to the LCD Animator. The CMD routine will be used exclusively (and extensively) by the user. Many references are made in the E-1330 programming guide with regard to sending a command to the E-1330; this is the subroutine that must be called for sending these commands.

DAT (Device, Status)

The DAT subroutine is required for sending data to the LCD Animator. The DAT routine will be used exclusively (and extensively) by the user. Many references are made in the E-1330 programming guide with regard to sending data to the E-1330; this is the subroutine that must be called for sending data and command parameters.

Example Subroutine Usage:

```
FRAME 0, 1 : Rem Tell LCDA Device 0 to go to frame 1.
FRAME 1, 32 : Rem Tell LCDA Device 1 to go to frame 32.
FRAME 7, 64 : Rem Tell LCDA Device 7 to go to frame 64.
LIGHT 0, 1 : Rem Tell LCDA Device 0 to turn on backlight.
LIGHT 1, 0 : Rem Tell LCDA Device 1 to turn off backlight.
LIGHT 7, 1 : Rem Tell LCDA Device 7 to turn on backlight.
'CLEAR GRAPHIC SCREEN ON DEVICES 0 THROUGH 7
For DV = 0 to 7
   CMD DV,$46 : Rem Set Cursor Position on Display 0.
   'POSITION THE CURSOR IN THE GRAPHIC SCREEN
   DAT DV,$00 : Rem Set LSB Position on Display.
   DAT DV,$02 : Rem Set MSB Position on Display.
   'Write some graphics to the screen.
   CMD DV,$42 : Rem Write to the screen
   For N = 1 to 4096
      DAT DV,$00 : Rem Turn Off Pixels in 4096 places.
   Next N
Next DV
```

Quick Overview of Subroutine Usage:

Speaking to several LCD Animators is just as easy as speaking to 1. The example outlined at left shows how to send simple commands to several displays sharing a single serial port. The core routines outlined above are required for this program to work.

The batteries must be attached to the battery backup system, and the LCD Animator Setup Utility must be used to initialize the display (a one-time process as long as good batteries stay attached to the LCD Animator). Note that FRAME can have a maximum value of 64 when in the ASCII text and graphics mode, and up to 128 when in the graphic-only mode.

NOTE: The following routines are commands that can be sent to the LCD Animator. Commands and data must be sent using CORE routines such as DAT and CMD, which subsequently call the subroutines CD and SENDBYTE.

Programming the LCD Animator:

Now that the core routines have been ported, you are free to do virtually anything you can think of just by using the correct subroutine call. For the most part, you will use the CMD, DAT, LIGHT, and FRAME subroutines to perform all LCD Animator operations.

To start with, you may want to know a couple of the commands that will be useful when interfacing the LCD Animator with primitive microcontrollers. Under normal operation, you would use the Windows 95 software to load in your graphics and display information. You can use the SLEEP command to divert the power for memory to the battery backup system.

```
'Call the CMD Subroutine with a
'Device Number of 0 to send a command.
'The following statement puts LCD Animator
'at address 0 to sleep (command 83 decimal)
CMD 0,83
```

SLEEP

The SLEEP command puts the display in sleep mode. When sleeping, the memory (including text, graphics, and display information) is kept in nonvolatile memory if backup batteries are attached. The LCD Animator will continue to service the backlight (if turned on) and both interface processors, and it will be safe to remove power from the controller (but not the battery backup). Make sure the correct device number is given when calling the CMD subroutine. Also note that 83 is in decimal (not HEX) form.

NOTE: The LCD Animator may be put to sleep by connecting the 2 SLEEP pins together (G and S). Using the hardware sleep mode, the LCD Animator will not wake up until the next powerup. Note that batteries MUST BE CONNECTED for sleep mode to work properly. We STRONGLY RECOMMEND removing the 12 volt power supply from the LCD Animator while in sleep mode.

```
'Send the Wake Command to device 0.
'Note that Parameters are in HEX.
Device=0
CMD Device, $40 : Rem Send System Set Command
DAT Device, $38 : Rem Send P1 of System Set
CMD Device, $59 : Rem Send Display On Command
CMD Device, $4C : Rem Set Cursor Right Moving
```

WAKE

The following subroutine calls will wake the LCD Animator. The LCD Animator automatically wakes when powered, displaying the last frame seen on the screen before the SLEEP command was issued (if power was not removed from the controller) or the first frame in memory if power was removed. The WAKE command is provided, should you need it in your application, but it should not be necessary.

NOTE: The software-issued WAKE command works only after the software-issued SLEEP command. The WAKE command will not work if the display was put to sleep using the SLEEP jumper located on the LCD Animator controller board.

The E-1330 manual (shipped with the LCD Animator on disk) is loaded with other cool things you can do with the CMD and DAT routines. These routines can be used to custom configure the LCD Animator, scroll images, and much, much more. To read this document, you will need Adobe Acrobat Reader Version 3.0 or later.

When power is applied to the LCD Animator, 3 on-board processors will begin to boot up. You should wait at least 1 second before sending data to the LCD Animator, as it is not ready for operation until the processors have completed their startup cycle, approximately 1 second.

Screen Organization and Setup:

Before we go any further, we need to have a quick lesson on the LCD screen and how it is organized. The included Windows 95 program "LCD Animator Setup Utility" loads the LCD Animator with screen configuration information. This initialization lays the foundation for all LCD Animator operations.

The cursor can be positioned anywhere on the text or graphics screen. If it is positioned on the text screen, it will be displayed as a small flashing box. If it is positioned on the graphics screen, it will be invisible. Regardless of whether it is in the graphics or the text screen, the function of the cursor is the same ... to tell where in memory the next byte of data will be written.

In memory, the text screen is located above the graphics screen as shown in Figure 1. Positioning the cursor at the beginning of memory places the text cursor in the upper left corner of the text screen. As text is written to the screen, the cursor automatically moves from left to right and from top to bottom. The text screen can hold 512 ASCII characters (32 columns x 16 lines).

If you accidentally write too many characters to the text screen, the cursor will automatically move into the graphics screen. Your ASCII text characters will then appear as pixels at the top of the graphics screen. Care should be taken to ensure text is not written beyond $01FF. Note that while the text screen is actually above the graphics screen in memory, the graphics and text screens appear overlaid on top of each other. This allows you to independently manipulate the text and graphic layers. See Figure 1 below.

Moving the Cursor:

Now, with the screen organization under our belt, we can begin to use the LCD Animator. The most common thing you are going to want to do as a programmer is move the cursor around on the screen and write some text with a few graphics.

The E-1330 command for positioning the cursor is $46 (hex). To send the cursor position command to the E-1330, you need to use the CMD subroutine.

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
```

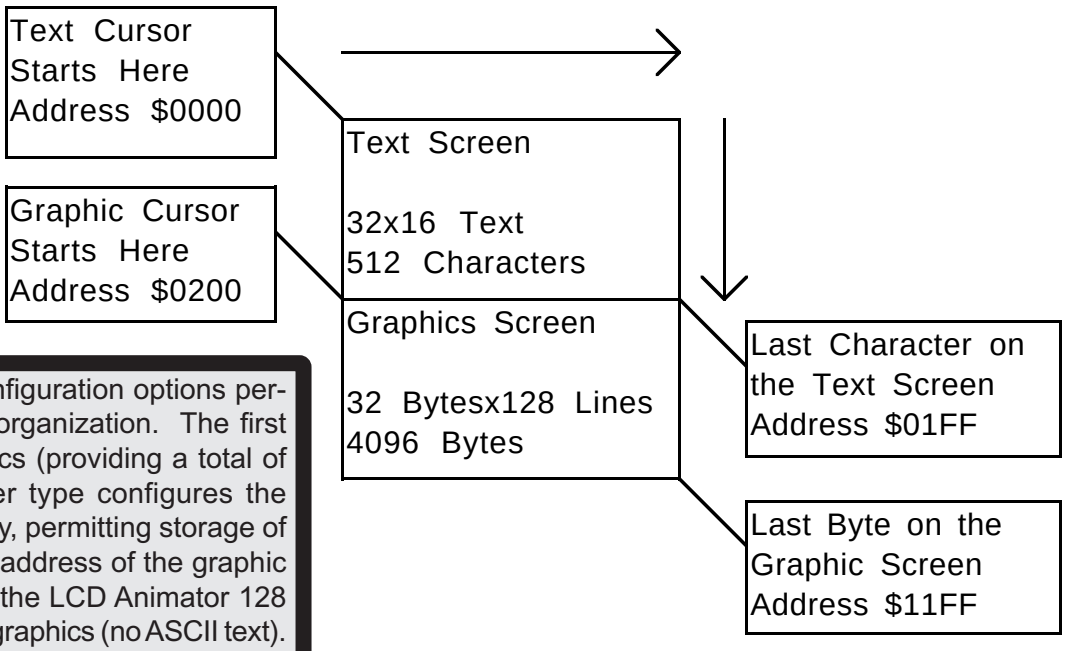
The command $46 requires 16 bits of position information to accurately position the cursor. Always use the DAT subroutine when sending data to the E-1330. So, if you want to place the cursor in the upper left corner (placing the cursor at the beginning of the text screen), use the following:

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$00 : Rem Set MSB Position on Display.
```

If you want to place the cursor at the beginning of the graphics screen, use the following command (see address of the beginning of the graphics screen in Figure 1 for where the DAT came from):

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$02 : Rem Set MSB Position on Display.
```

LSB = Least Significant Byte of a 16-Bit Word.
MSB = Most Significant Byte of a 16-Bit Word.


Text Cursor Starts Here Address $0000
Graphic Cursor Starts Here Address $0200
Text Screen
32x16 Text
512 Characters
Graphics Screen
32 Bytesx128 Lines
4096 Bytes
Last Character on the Text Screen Address $01FF
Last Byte on the Graphic Screen Address $11FF

IMPORTANT: Two configuration options permit 2 types of screen organization. The first type is text and graphics (providing a total of 64 frames). The other type configures the screen for graphics only, permitting storage of 128 frames. The start address of the graphic screen is $0000 when the LCD Animator 128 is set for 128 frames of graphics (no ASCII text).

Figure 1: LCD Screen Organization for 256x128 Displays.

Writing Text to the Display:

Now that we have successfully positioned the cursor, it's time to write text to the screen. Writing to the screen requires the use of another E-1330 function, command $42.

```
'Position the cursor in the text screen
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$00 : Rem Set MSB Position on Display.

'Write some text to the screen.
CMD 0,$42 : Rem Write to the screen
DAT 0,$41 : Rem Write the letter A to the screen.
DAT 0,$42 : Rem Write the letter B to the screen.
DAT 0,$43 : Rem Write the letter C to the screen.
DAT 0,$44 : Rem Write the letter D to the screen.
DAT 0,$45 : Rem Write the letter E to the screen.
DAT 0,$46 : Rem Write the letter F to the screen.
DAT 0,$47 : Rem Write the letter G to the screen.
```

```
'Position the cursor in the text screen
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$00 : Rem Set MSB Position on Display.

'Write All Characters to the screen.
CMD 0,$42 : Rem Write to the screen
For N = 0 to 255
     DAT 0, N : Rem Write Character N to the Screen.
Next N
```

Writing Graphics to the Display:

Now, let's write a few bytes of graphics to the display. Don't forget, we have to position the cursor on the graphic screen before writing graphics.

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$02 : Rem Set MSB Position on Display.

'Write some graphics to the screen.
CMD 0,$42 : Rem Write to the screen
DAT 0,$FF : Rem Write 11111111 to the screen.
DAT 0,$00 : Rem Write 00000000 to the screen.
DAT 0,$F1 : Rem Write 11110001 to the screen.
```

Lost?

Probably the most difficult concept to grasp at this point is the LSB/MSB position information. Here is an example of how it works:

Say you want to position the cursor in the middle of the graphics screen. Well, there are a total of 4096 positions the graphics cursor can be located in ... so the middle position would be 2048.

Somehow, we have to tell the controller we want to move to position 2048. To do this, a 16-bit word is needed. Sending a 16-bit word consists of sending 2 bytes to the display. The first byte tells the cursor to move in increments of 1 byte. The second byte tells the controller to move the cursor in increments of 256 bytes.

So moving the cursor to the middle position would require us to send 256x8 to reach the 2048th position (center of the screen). Here is what our transmission would look like:

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$08 : Rem Set MSB Position on Display.
```

Similarly, if you wanted to position the graphics cursor in the 2050th position, send the following:

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$02 : Rem Set LSB Position on Display.
DAT 0,$08 : Rem Set MSB Position on Display.
```

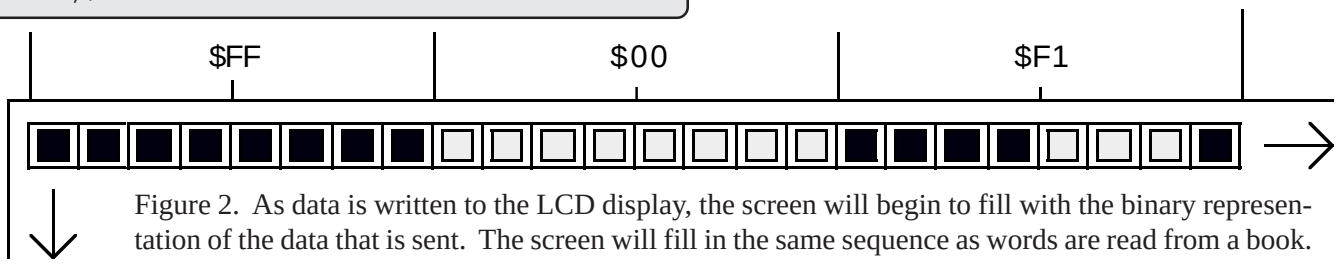


Figure 2. As data is written to the LCD display, the screen will begin to fill with the binary representation of the data that is sent. The screen will fill in the same sequence as words are read from a book.

Example: Clear Graphics Screen

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$02 : Rem Set MSB Position on Display.

'Write some graphics to the screen.
CMD 0,$42 : Rem Write to the screen
For N = 1 to 4096
   DAT 0,$00 : Rem Turn Off Pixels in 4096 places.
Next N
```

Example: Fill Graphics Screen

```
CMD 0,$46 : Rem Set Cursor Position on Display 0.
DAT 0,$00 : Rem Set LSB Position on Display.
DAT 0,$02 : Rem Set MSB Position on Display.

'Write some graphics to the screen.
CMD 0,$42 : Rem Write to the screen
For N = 1 to 4096
   DAT 0,$FF : Rem Turn On Pixels in 4096 places.
Next N
```

Programming Examples:

The following examples REQUIRE the core subroutines outlined earlier in this guide. Using the core routines will significantly reduce redundant programming and will ensure your software stays NCD RS-232 network compliant.

```
Sub TEXT(Device, Xpos, Ypos, TXT$)

    'Send SET CURSOR POSITION Command
    CMD Device, &H46

    'Determine Cursor Position on Text Screen
    CPOS = (Xpos - 1) + ((Ypos - 1) * 32)

    ‘Send the Cursor Position Data
    If CPOS <= 255 Then
       DAT Device, CPOS
       DAT Device, 0
    Else
       DAT Device, CPOS - 256
       DAT Device, 1
    End If

    'Issue the command for writing to the display
    CMD Device, &H42

    'Send Text to Display
    For AA = 1 To Len(TXT$)
       DAT Device, Asc(Mid$(TXT$, AA, 1))
    Next AA

End Sub
```

TEXT (Device, X, Y, Text$)

The TEXT routine shown at left will allow you to position text anywhere on the LCD screen. The DEVICE number refers to the LCD Animator that you want to send text to. Valid device numbers range from 0 to 7.

The X and Y parameters are used to position the text on the screen. X defines the starting column and has a valid range of 1 to 32. Y defines the starting line and has a valid range of 1 to 16. TXT$ holds a string of text to be transmitted.

The routine begins by sending the command for positioning the cursor by sending 46 hex. CPOS holds the linear position data using a simple equation to convert X/Y coordinates into a linear (1 to 512) position. The data transmissions send the linear position information (1 to 512) as a 16-bit word using 2 8-bit transmissions.

42 hex tells the LCD Animator to direct all incoming data directly to the screen. As data is received, the cursor automatically shifts to the right, displaying each character of text.

```
Sub SPEED [Device, File$]

   'Issue the command to set the cursor position
   CMD Device, &H46
   'Set the cursor position to the beginning of
   ‘the graphic screen
   DAT Device, &H0
   DAT Device, &H2

   'Issue the write command
   CMD Device, &H42

   'Open the Speed file
   Open File$ For Input As 1
   'Read the header
   Input #1, a

   'Count through the length of the file
   For n = 1 To 4096
      'read the data
      Input #1, a
      'send the data to the display
      DAT Device, a
   Next n

   'Close the file
   Close #1
End Sub
```

SPEED (Device, Frame, File$)

The LCD Animator Setup Utility allows you to create speed files. A speed file is simply a BMP image that has been converted to an ASCII text file. A full-screen image on a 256x128 graphics display occupies 4096 bytes. Similarly, a speed file has 4096 entries that represent every byte of a full-screen image, plus a header entry that identifies the size of the image. Speed files are the fastest and easiest type of image files to send to the LCD Animator, especially if you need to store images in a primitive microcontroller or DOS environment. The example shown at left will upload a speed file directly to the LCD screen.

The DEVICE parameter is a number from 0 to 7, identifying which LCD Animator is to receive the image. The FILE$ parameter refers to the speed file to transmit. Use the complete path when specifying a file to upload.

The SPEED subroutine is very similar to the TEXT subroutine outlined above. The text subroutine positions the cursor on the text screen, while the SPEED subroutine positions the cursor on the graphics screen.

Programming Tutorial

The following QBasic program (found on installation disk 2 A:\QBasic\LCDA64.BAS) demonstrates usage of the outlined routines in a real-world application. This tutorial may be found in the QBasic folder on Installation Disk 2. Note that files are loaded from disk 2, so keep the tutorial in the A drive. Also note that QBasic should be run from DOS, NOT Windows. Running QBasic from Windows will cause errors in the RS-232 transmission.

The tutorial makes use of SPEED files. SPEED files are created using the LCD Animator Setup Utility, and they contain raw image data that can be dumped to the display. SPEED files can be used by any computer or microcontroller using ANY language. SPEED files can be identified by their .SPD extension. Open one of the SPEED files in your favorite text editor to examine the contents. You will find a header and 4096 data entries corresponding to 4096 bytes that can be uploaded to form a full-screen image on a 256x128 graphic display (256 ÷ 8 bits = 32 columns. 32 x 128 lines = 4096 bytes).

The following program was designed to work with the LCD Animator 128 using the 64-frame text and graphic mode. See the program changes on the next page for using the graphic-only 128-frame system.

```
DECLARE SUB WATE (tim!)
DECLARE SUB FRAME (Device!, FRAM!)
DECLARE SUB CLRGRA (Device!)
DECLARE SUB SPED (Device!, File$)
DECLARE SUB CLRTXT (Device!)
DECLARE SUB TEXT (Device!, XPos!, YPos!, TXT$)
DECLARE SUB LIGHT (Device!, Status!)
DECLARE SUB Wake (Device!)
DECLARE SUB CD (Device!, Status!)
DECLARE SUB SENDBYTE (Device!, Byte!)
DECLARE SUB CMD (Device!, Byte!)
DECLARE SUB DAT (Device!, Byte!)

'Open the RS-232 Port for Communications
OPEN "COM2:19200,N,8,1,CD0,CS0,DS0,OP0" FOR OUTPUT AS #1
Device = 0:         REM Set the Software Device Number
LIGHT Device, 1:    REM Turn On (1) Backlight of device 0
Wake Device:        REM Wake up device 0

FRAME Device, 1:    REM Set the Current Frame Number to 1
CLRTXT Device:      REM Clear the Text Frame
CLRGRA Device:      REM Clear the Graphic Frame

'Write text starting at column 1 of line 4
TEXT Device, 4, 1, "Another Great Product From"
TEXT Device, 5, 3, "National Control Devices"
TEXT Device, 7, 14, "The LCD Animator 128"
TEXT Device, 10, 16, "By Ryan Sheldon"

'Upload a SPEED file graphic to Display
'created by the LCD Animator Setup Utility
SPED Device, "A:\BMP\001.SPD"

FRAME Device, 2:    REM Set the Current Frame Number to 2
CLRTXT Device:      REM Clear the Text Frame
CLRGRA Device:      REM Clear the Graphic Frame

'Write text starting at column 1 of line 4
TEXT Device, 1, 1, "Presenting Dinosaurs that Bite"
'Upload a SPEED file graphic to Display
'created by the LCD Animator Setup Utility
SPED Device, "A:\BMP\016.SPD"

'Switch Between Frames of the Animation
DO
  FRAME Device, 1
  WATE 60
  FRAME Device, 2
  WATE 60
LOOP

'CMD Device, 83:    REM Send the sleep command, Display will go blank
CLOSE #1
```

QBasic automatically declares subroutines. Many newer languages do not require declarations.

Note that the software device number is 0. Jumpers J1, J2, and J3 must be removed on the LCD Animator.

When exiting the LCD Animator Setup Utility, the device is put to sleep and is activated with the WAKE subroutine.

CLRTXT and CLRGRA clear the text and graphic layers. CLRGRA is not required since the uploaded image will replace the garbage on the screen, but an example subroutine call is shown.

The SPED subroutine uploads a SPEED file directly to the display. Use the FRAME subroutine to set which frame the image is loaded into.

Here, a second frame is loaded with text and graphics. Once uploaded, using the FRAME subroutine will switch between the 2 frames on the display.

The TEXT subroutine is used to precisely position text on the display. This routine is surprisingly simple to use and port.

Here, the frames are switched. The example shows switching between frames 1 and 2. Valid frame changes can be from 1 to 64. Since frames 3-64 are unloaded, garbage will be displayed. The delay permits frame changes in about 2-second intervals.

This program never makes it down here, but if it did, the LCD Animator would be put to sleep and the CLOSE #1 would release the serial port.

```
SUB CD (Device, Status)
  'Compute Device Number
  DEV = (Device + 8) * 16
  IF Status = 1 THEN
    'Turn On CD Line
    PRINT #1, CHR$(DEV + 3);
  ELSE
    'Turn Off CD Line
    PRINT #1, CHR$(DEV + 2);
  END IF
END SUB

SUB CLRGRA (Device)
  'Home the Cursor in the Graphic Screen
  CMD Device, &H46
  DAT Device, &H0 :         REM A
  DAT Device, &H2 :         REM B
  'Issue the write command
  CMD Device, &H42
  'Send blank data to the screen
  FOR n = 0 TO 4095
    DAT Device, &H0
  NEXT n
END SUB

SUB CLRTXT (Device)
   'Compute Device Number
   DEV = Device * 16
   'Home the Cursor
   CMD Device, &H46
   DAT Device, 0 :          REM C
   DAT Device, 0 :          REM D
   'Issue the Write Command
   CMD Device, &H42
   'Send Text to the Display 1 character at a time
   FOR aa = 1 TO 512
     DAT Device, 32: REM Send SPACE character
   NEXT aa
END SUB

SUB CMD (Device, Byte)
   'Turn On CD Line of Display to Put in Command Mode3
   CD Device, 1
   'Send a Data Byte
   SENDBYTE Device, Byte
END SUB

SUB DAT (Device, Byte)
  'Turn Off CD Line for Data Mode of Display
  CD Device, 0
  'Send Data Byte
  SENDBYTE Device, Byte
END SUB

SUB FRAME (Device, FRAM)
  'Compute the Device Number
  DEV = (Device + 8) * 16
  'FR Can Equal 0 to 15 Only
  FR = FRAM - 1: Chapter = 1
  'Count Up the Chapters in
  'Increments of 16 Pages
  WHILE FR > 15
    Chapter = Chapter + 1
    FR = FR - 16
  WEND
  'Send the Page Change Command
  PRINT #1, CHR$(DEV + 4)
  'Send the Page to Go to (FR=0 to 15)
  PRINT #1, CHR$(DEV + FR)
  'Send the Chapter Change Command with Chapter
  PRINT #1, CHR$(DEV + 5 + Chapter)
END SUB

SUB LIGHT (Device, Status)
  'Compute Device Number
  DEV = (Device + 8) * 16
  IF Status = 1 THEN
    'Turn On Backlight
    PRINT #1, CHR$(DEV + 1);
  ELSE
    'Turn Off Backlight
    PRINT #1, CHR$(DEV + 0);
  END IF
END SUB

SUB SENDBYTE (Device, Byte)
  'Compute Device Number
  DEV = Device * 16
  'Compute Least Significant Nibble
  LSN = (Byte AND 15)
  'Compute Most Significant Nibble
  MSN = (Byte AND 240) / 16
  'Send Least Significant Nibble
  PRINT #1, CHR$(LSN + DEV);
  'Send Most Significant Nibble
  PRINT #1, CHR$(MSN + DEV);
END SUB
```

The CD subroutine is NEVER called by the user. This subroutine is used exclusively by the CMD and DAT subroutines. The CD subroutine controls the COMMAND or DATA mode of the LCD Animator. Command mode can be used to position text and graphics, while data mode consists of actual ASCII text and graphics.

The CLRGRA is a simple routine to clear the graphic screen of the currently selected frame of the LCD Animator. There are 4096 bytes that occupy the 256x128 graphic screen; therefore, 0 (&H0) must sent 4096 times to clear the screen. Replacing the &H0 with &HFF will fill the screen with black. All data values are given in HEX signified by &H. Decimal values of 0 to 255 are also permitted, depending on the preferences of the programmer.

Much like the CLRGRA subroutine, the CLRTXT subroutine is responsible for clearing the text layer of the LCD Animator. This routine begins by placing the cursor at the beginning of the text screen (see REM C&D). The beginning of the graphic screen is at REM A&B. There are 512 characters that occupy the 256x128 display (32x16); therefore, the space character (decimal 32) must be sent 512 times to clear the text screen.

The CMD and DAT subroutines are called exclusively by the user and other higher level routines. Using only the CMD and DAT subroutines, the LCD Animator can perform ANY text or graphic operation. Backlight and frame-change commands cannot be accessed by the CMD and DAT subroutines.

Think of the LCD Animator as a book with a maximum of 8 chapters and 16 pages per chapter. There are only 4 chapters available when in graphic AND text mode. The graphics-only mode permits 8 chapters. FRAME changes are accomplished by converting a number from 1 to 128 to its coresponding chapter and page. The RAMPAGE chip is responsible for setting one of 16 pages while the LCD Animator Network Interface Processor (NIP) changes the chapter.

The backlight on the LCD Animator is automatically turned on upon powerup and off when put to sleep. In addition, the user has full software control of the backlight under normal operating conditions. Be warned that the LCD Animator produces high voltages to drive the EL inverter. Avoid contact with terminals that connect to the backlight.

SENDBYTE is used exclusively by the CMD and DAT subroutines. This routine is responsible for formatting a byte of data for compliance with the NCD RS-232 control network. Two transmissions are required to send a single byte to the display.

```
SUB SPED (Device, File$)
  'Home the Cursor in the Graphic Screen
  CMD Device, &H46
  DAT Device, &H0
  DAT Device, &H2
  'Issue the write command
  CMD Device, &H42
  'Open Data File
  OPEN File$ FOR INPUT AS #2
  INPUT #2, A$: REM Read the header and discard contents
  'Read the rest of the file
  FOR n = 0 TO 4095
     INPUT #2, A:    REM Read Data Byte from File
     DAT Device, A:  REM Send Data Byte to Display
  NEXT n
  'Close the Data File
  CLOSE #2
END SUB
```

The SPED subroutine dumps a SPEED file to the display. Note the similarities between this subroutine and the CLRGRA subroutine. The only real difference is that instead of sending 0 to clear a byte, data is read from a data file and dumped to the display to form an image.

```
SUB TEXT (Device, XPos, YPos, TXT$)
   'Compute Device Number
   DEV = Device * 16
   'Setup Display for Text Positioning
   CMD Device, &H46
   'Determine Linear Cursor Position on the Text Screen
   CPos = (XPos - 1) + ((YPos - 1) * 32)
   'Position the Cursor on the Display
   IF CPos <= 255 THEN
     DAT Device, CPos
     DAT Device, 0
   ELSE
     DAT Device, CPos - 256
     DAT Device, 1
   END IF
   'Issue the Write Command
   CMD Device, &H42
   'Send Text to the Display 1 character at a time
   FOR aa = 1 TO LEN(TXT$)
      DAT Device, ASC(MID$(TXT$, aa, 1))
   NEXT aa
END SUB
```

The TEXT subroutine is most similar to the CLRTXT subroutine, excep that the CPOS equation converts X/Y coordinates into linear (0 to 511) coordinates. Once the position is set, the subroutine counts from 1 to the length of the incoming text. Instead of continuously sending the space character (decimal 32), ASCII values of the character in the string are sent.

Before the LCD Animator will respond to anything, it must receive the WAKE command. Note that this command is automatically sent upon powerup of the controller.

```
SUB Wake (Device)
   'Send Wake up commands to the LCD Animator
   CMD Device, &H40: REM System Set Command
   DAT Device, &H38: REM First Parameter of the System Set Command
   CMD Device, &H59: REM Turn the Display On
   CMD Device, &H4C: REM Tell Cursor to Move Right
END SUB
```

WATE is a very dumb delay routine.

```
SUB WATE (tim)
  FOR Delay = 1 TO tim * 3000
  NEXT Delay
END SUB
```

Graphics Only Mode Programming:

A few modifications must be made to the above program when using the LCD Animator in its 128 frame graphic only mode. Since this mode does NOT support text, all text related subroutines may be deleted from your program. The only change that is required is the start address of the graphic pointer while in this mode. So all graphic related functions must be changed to the following:

```
SUB CLRGRA (Device)
  'Home the Cursor in the Graphic Screen
  CMD Device, &H46
  DAT Device, &H0 :        REM A
  DAT Device, &H0 :        REM B
  'Issue the write command
  CMD Device, &H42
  'Send blank data to the screen
  FOR n = 0 TO 4095
    DAT Device, &H0
  NEXT n
END SUB
SUB SPED (Device, File$)
  'Home the Cursor in the Graphic Screen
  CMD Device, &H46
  DAT Device, &H0 :        REM A
  DAT Device, &H0 :        REM B
  'Issue the write command
  CMD Device, &H42
  'Open Data File
  OPEN File$ FOR INPUT AS #2
  INPUT #2, A$: REM Read the header and discard contents
  'Read the rest of the file
  FOR n = 0 TO 4095
     INPUT #2, A:    REM Read Data Byte from File
     DAT Device, A:  REM Send Data Byte to Display
  NEXT n
  'Close the Data File
  CLOSE #2
END SUB
```

The only difference in these routines is REM A. Compare to other REM A marks in previous subroutines.

The FRAME subroutine takes any frame number from 1 to 128. You should only use 1 to 64 while in the graphic/text mode and 1 to 128 while in the graphics-only mode. Using frames 65-128 will repeat frames 1-64 while in the graphic/text mode.

Programming Techniques:

The LCD Animator Setup Utility employs all routines discussed in this guide to provide a user-friendly interface to the LCD Animator. Future versions will support additional features, LCD Animator models, and LCD display sizes. By now, you should have a very good idea about what the LCD Animator is capable of doing, and hopefully, how it can be used in your application.

Development of this device took many months, nearly a year altogether. And during this time, we have encountered about every conceivable problem ... and even better, a few shortcuts that just might make your life a little easier.

First of all, a delay will be required at 38.4K and 115.2K baud when using the frame change command. This delay gives the LCD Animator enough time to communicate to the RAMPAGE chip. Please see the following example:

```
SUB FRAME (Device, FRAM)
  'Compute the Device Number
  DEV = (Device + 8) * 16
  'FR Can Equal 0 to 15 Only
  FR = FRAM - 1: Chapter = 1
  'Count Up the Chapters in
  'Increments of 16 Pages
  WHILE FR > 15
    Chapter = Chapter + 1
    FR = FR - 16
  WEND
  'Send the Page Change Command
  PRINT #1, CHR$(DEV + 4)
  'Send the Page to Go to (FR=0 to 15)
  PRINT #1, CHR$(DEV + FR)
'PUT A DELAY HERE. TRY 7 OR 14 µS.
  'Send the Chapter Change Command with Chapter
  PRINT #1, CHR$(DEV + 5 + Chapter)
END SUB
```

A general rule to follow is to write text before graphics. It's a simple trick to eliminate the flash of the text cursor. Remember, the graphic and text cursors are identical; their function is defined only by their location in memory. When in the text area, a cursor will be displayed. When in the graphic area, a cursor will not be shown. Moving the cursor is as simple as sending the position command and the location to move to:

```
CMD Device, &H46 :     REM Send Cursor Position Command
DAT Device, &H0 :      REM Send Location (Part 1, LSN)
DAT Device, &H0 :      REM Send Location (Part 2. MSN)
```

The above example sets the cursor on the text screen while in text/graphics mode, but also sets the location to the beginning of the graphic screen while in the 128-frame graphic-only mode.

Another cool trick is to flash the software-controlled backlight. This is a great way to get attention in store displays and emergency display applications.

Consider using Adobe Priemiere to create animations for the LCD Animator. Quicktime movies can be saved as individual frames that can be easily loaded into the LCD Animator Setup Utility. Avoid rapid-changing images in favor of soft and slow transitions.

Take advantage of the high frame rate of 960 frames per second. Impressive visual effects can make all the difference in how your project is perceived by others.

Advanced E-1330 Programming:

Adventurous programmers may want to experiment with other E-1330 commands such as flash, scroll, and cursor direction. The E-1330 technical manual can be found on the internet at http://www.smos.com/.

To read this document, you must have Adobe Acrobat Version 3.0. This is a free program available from Adobe at http://www.adobe.com.

On page 122 of the E-1330.PDF file, an application note is outlined, showing you how to configure the E-1330 for a 512x128 LCD graphic display.

Configuring the E-1330 as outlined in the text is very simple. Turn to page 123 to see how commands are translated from the E-1330 to the LCD Animator:

In the "No." column, items 1 and 2 refer to powerup. No commands will be required for these phases. Item 3 refers to sending the SYSTEM SET command, which (along with its parameters) will configure the controller for any size graphic display (except LCD Animator 128). The command SYSTEM SET is issued by sending $40 hex (see "Command" column C=40H).

This command can be sent to the LCD Animator using the CMD core subroutine as follows:

```
CMD 0, &H40 : Rem Send SYSTEM SET Command to device 0.
```

The SYSTEM SET command has 8 parameters that must also be sent to the LCD Animator. Parameters for any E-1330 command are transmitted using the DAT core subroutine. For example:

```
DAT 0, &H38 : Rem Send Parameter P1 to device 0.
DAT 0, &H87 : Rem Send Parameter P2 to device 0.
DAT 0, &H07 : Rem Send Parameter P3 to device 0.
DAT 0, &H3F : Rem Send Parameter P4 to device 0.
DAT 0, &H49 : Rem Send Parameter P5 to device 0.
DAT 0, &H7F : Rem Send Parameter P6 to device 0.
DAT 0, &H80 : Rem Send Parameter P7 to device 0.
DAT 0, &H00 : Rem Send Parameter P8 to device 0.
```

Similarly, the SCROLL command must be sent along with its parameters to the LCD Animator as follows:

```
CMD 0, &H44 : Rem Send SCROLL Command to device 0.
DAT 0, &H00 : Rem Send Parameter P1 to device 0.
DAT 0, &H00 : Rem Send Parameter P2 to device 0.
DAT 0, &H40 : Rem Send Parameter P3 to device 0.
DAT 0, &H00 : Rem Send Parameter P4 to device 0.
DAT 0, &H10 : Rem Send Parameter P5 to device 0.
DAT 0, &H40 : Rem Send Parameter P6 to device 0.
DAT 0, &H00 : Rem Send Parameter P7 to device 0.
DAT 0, &H04 : Rem Send Parameter P8 to device 0.
```

Note that HEX notation will differ between programming languages. Some will require a "$" preceeding the hex value, others will require an H after the hex value. It is important to send HEX notation properly for the programming language you will be using.

The NCD RS-232 Control Network:

The LCD Animator was designed to comply with the NCD RS-232 control network. Up to 16 NCD devices can be controlled from a single serial port using this protocol. With the release of the QNET adapter, up to 64 NCD devices will be controllable from a single serial port. The LCD Animator counts as 2 devices on this network.

The LCD Animator can share a serial port with our audio/video switchers, relay drivers, motor controllers, audio signal processors, A/D and I/O boards, and much more. NCD products can be found in the most demanding environments, from industrial automation and manufacturing to the day-to-day hobbyest project. The back page of this manual will give you an idea of what can be accomplished with this network.

Above all, the NCD control network was designed to be simple, affordable, and expandable. Many products found in the NCD product line directly compete with manufacturers charging hundreds, even thousands more. Our expandable network gives you more power from a single serial port than most users will ever need.

We are continuously developing new products. We choose our projects very carefully. Our goal is to provide users with unique computer control products that simply do not exist in the marketplace. At the same time, we try to fill a need for making the products that do exist, compatible with all our line. Compatibility is the key to our progression. A control network that is powerful can only be built on a foundation that is solid, diverse, and compliant to the needs of all our users.

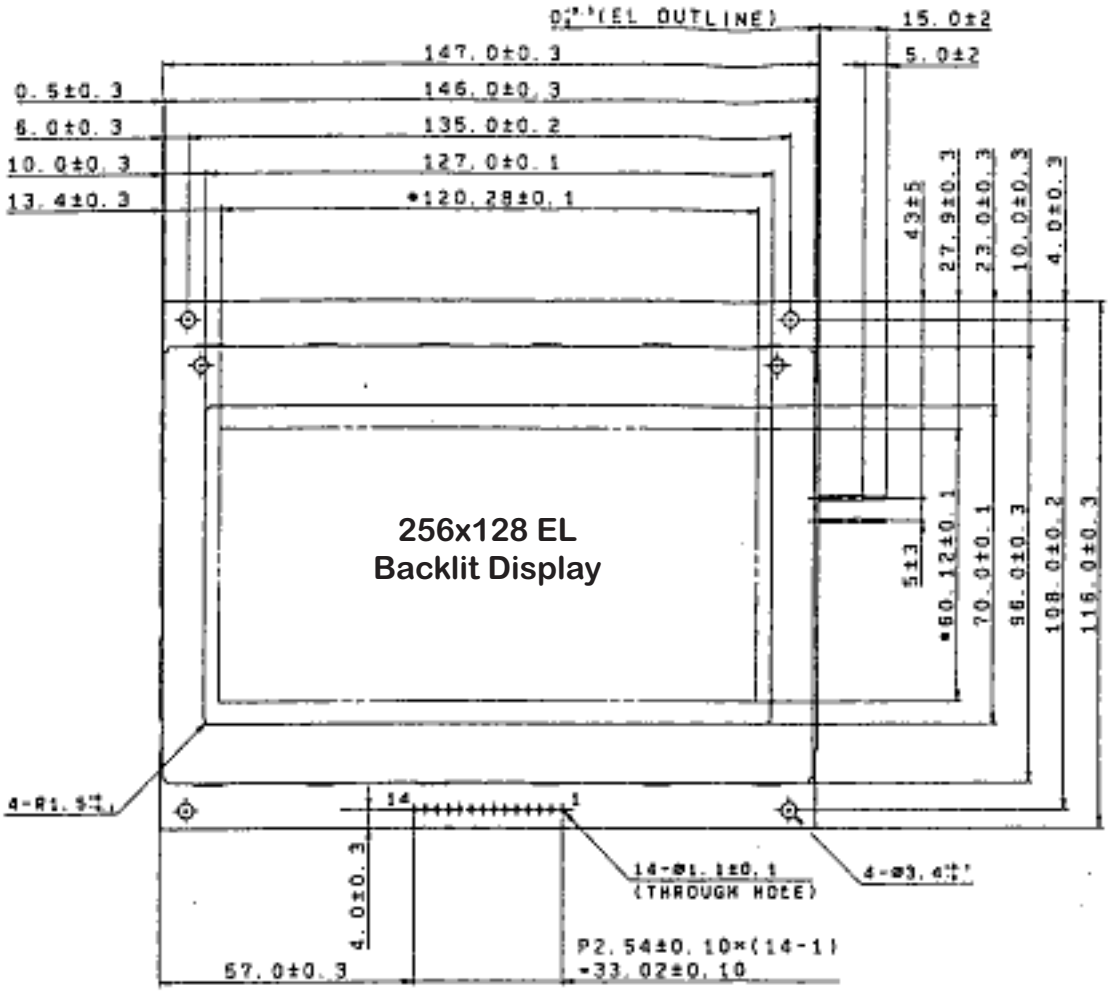
Closing:

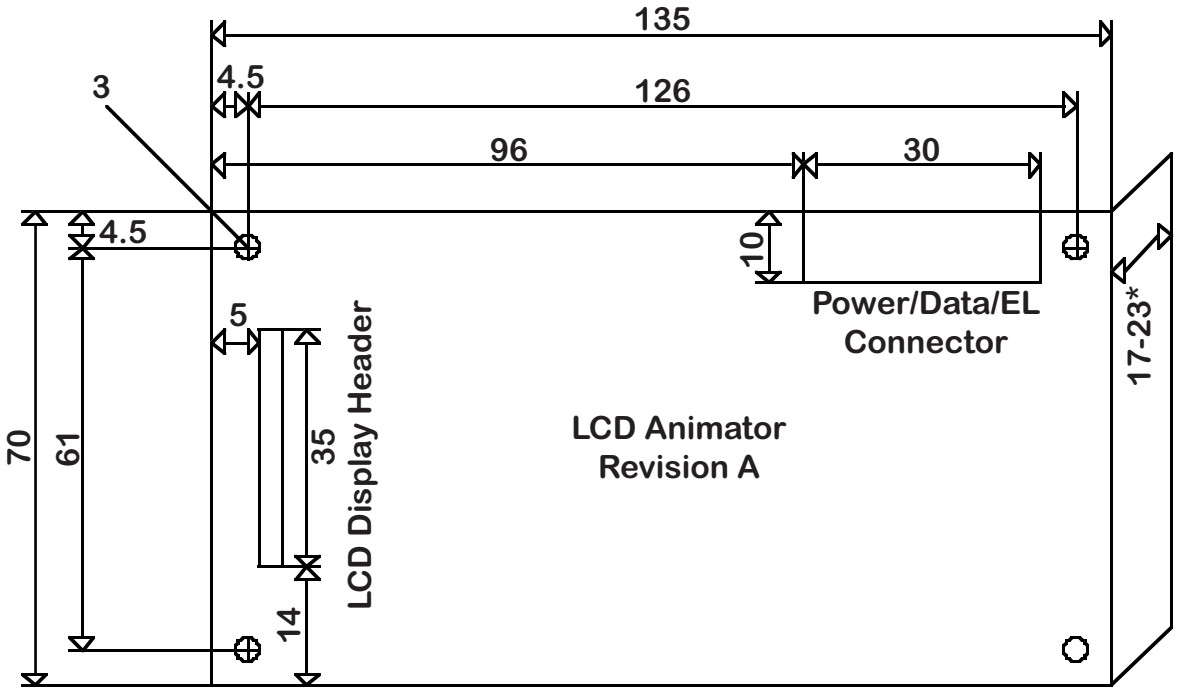
Have fun using the LCD Animator. I sincerely hope you have as much fun using it as I had designing it. This project has taken months to complete, but I think you will find it a powerful tool. Don't forget to keep me posted on what you are doing with it ... and e-mail or snail-mail me pictures of the LCD Animator in your project ... we'll add them to our web page.

If you have any questions, please call me at (404) 244-2432 or e-mail me at ncdryan@aol.com.

Thanks again for helping to make the LCD Animator a successful product. This encourages us to make other new and innovative products.

Ryan Sheldon
National Control Devices
(404) 244-2432
ncdryan@aol.com





All dimensions are in millimeters.
*Height Depends on Model of EL Inverter Installed.

Absolute Maximum Ratings

Parameter	Minimum	Typical	Maximum	Units
Logic Supply	9.0	12.0	14.0	Volts
Backlight Supply	4.5	5.0	5.5	Volts
Bias Generator	-	-8.6 (loaded)	-15 (unloaded)	Volts
LCD Logic Supply	4.9	5.0	5.1	Volts
EL Voltage	-	100	-	Volts
Standby (Sleep) Supply	2.0	-	5.5	Volts
Logic Supply Current	.059	.062	.075	Amps
Logic & Backlight Current	.121	.125	.135	Amps
EL Current	-	14.5	18.5	mA
Inverter Input Current	-	105	126	mA
LCD Logic Supply Current	-	-	5.5	mA
LCD Drive Current	-	-	4.0	mA
Standby (Sleep) Current	-	1.9	3	µA
EL Frequency	-	400	-	Hz
Operating Temperature	0	-	+50	˚C
Storage Temperature	-20	-	+60	˚C

WARNING: The LCD Animator automatically turns on the backlight on startup. Never touch the EL backlight connectors or wires while in operation.

WARNING: This device is NOT polarity protected. Precautions must be taken to prevent damaging reverse voltages.

NOTE: We STRONGLY recommend a 12 volt power supply for the logic circuits of the LCD Animator. The bias generator is driven from the logic side of the supply and requires at least 12 volts to produce a current strong enough to drive the 256x128 LCD.

NOTE: The battery backup system requires a minimum of 2 volts for proper operation. We recommend the use of 2 or 3 AA batteries connected in series. Also note that permanent damage may result if the batteries are connected backwards or deliver more than 5.5 volts.

NOTE: The LCD Animator and its related functions are tested prior to shipping.

Troubleshooting:

Most problems with using the LCD Animator are related to improper jumper settings or wire connections. We strongly advise you to carefully examine the diagrams BEFORE connecting. Many problems can be avoided by applying power to the LCD Animator BEFORE running the LCD Animator Setup Utility. If you experience COM errors when running the LCD Animator Setup Utility, PLEASE CONSULT THE MANUFACTURER OF YOUR COMPUTER, NOT NCD TECHNICAL ASSISTANCE. We are unable to help you correct COM-related problems on unfamiliar systems. The LCD Animator Setup Utility provides a COM- related error message ONLY when it cannot access your COM port. This error indicates that you have a configuration or hardware problem with the COM port you are trying to use. You MUST resolve this problem before continuing.

Windows 95 Installation Problems

The LCD Animator Setup Utility installation program may give you an "Unexpected Error, Quitting" during installation. To correct this problem, read the README.TXT file on the Windows 95 Patch Disk.

Power Supply Problems

We strongly advise the use of a computer-grade regulated power supply for the LCD Animator. Note that some power supplies have a minimum load requirement. These power supplies will not work with the LCD Animator attached if the LCD Animator is not pulling enough power out of the supply to turn it on. Try using a load resistor (see the documentation for your power supply) or a different supply. Make sure you attach power to the LCD Animator EXACTLY as indicated on the diagram. The LCD Animator can be damaged by reverse voltages or exceeding the maximum voltage ratings.

Power Sequencing and Stalling

Most stalling occurs because batteries have not been attached to the battery backup system. Don't forget to power the LCD Animator BEFORE running the LCD Animator Setup Utility. Also, the LCD Animator should never be switched on-off-on without a delay. There should be a minimum 15-second delay from on to off to on again. If the LCD Animator appears to be frozen with a blank screen, switch off the power and wait a minimum of 15 seconds before turning it back on. If the problem persists, unplug the backup batteries, turn off the LCD Animator, wait 15 seconds, and turn it back on again.

Jumper Settings

Keep in mind that jumper settings on the LCD Animator are critical. They not only control the device number, but they also control the baud rate, the flow of RS-232 data throughout the board, and a variety of other important functions. When in doubt, set the jumpers in their default position and run the LCD Animator Setup Utility to verify the board is properly receiving and processing data.

Battery Backup System

The battery backup system requires the addition of at least one 2 to 5 volt battery. Some LCD Animators may not function at all if a battery is not attached. Keep in mind that the LCD Animator Setup Utility writes important configuration information into the LCD Animator. If the batteries are disconnected, the configuration data will be lost. Also note that you should ALWAYS issue the SLEEP command via hardware or software BEFORE operating power is removed. This puts the graphic controller chip in low-power mode, saving the batteries.

Garbage Screen on Startup

On initial startup, the LCD Animator will display random noise and characters. This is normal operation. The garbage on the screen is a direct reflection of what is in SRAM ... which could be anything when you first apply power. Use the battery backup system and preload images to make sure the end-user never sees this noise.

If preloaded images appear corrupt on startup, there is a problem with the battery backup system. Most likely, the LCD Animator was NOT properly put to sleep before power was removed. Note that the hardware SLEEP jumper will put the LCD Animator to sleep only when power is available. Power may then be removed after the display goes to sleep.

Blank Screen

If the LCD Animator fails to respond to initialization commands, test the voltage of the battery connected to B1+. If the backlight comes on for your display but the screen stays blank, try adjusting the contrast knob. The screen should turn black when turned to the extreme right. If you do NOT get a black screen, verify that the display is properly attached to the controller. Also note that a blank screen is a symptom of power on, power off, and power on again without a minimum 15-second delay. A blank screen may also indicate sleep mode.

Communication-Related Problems

The LCD Animator works well with 1 stop bit at data rates up to 19.2K baud. Use 2 stop bits at higher baud rates. See the FRAME subroutine on page 11 for additional delay requirements at 115.2K baud.

Device Number Problems

The remaining software problems are usually device number related. Make sure the device number in software matches the device number on the LCD Animator hardware. Keep in mind that the LCD Animator was designed to ignore data if it is not sent to the correct device number. If the LCD Animator appears to be ignoring data, check the device number on the board as well as in software. We STRONGLY recommend that you port the core subroutines EXACTLY as they are outlined in this guide. Don't try to take shortcuts. The core routines were engineered to match the needs of the LCD Animator.

Development-Related Problems

Other problems with regard to using the LCD Animator may strictly be software related. Always run the LCD Animator Setup Utility to ensure the display is working correctly and all wires are properly attached BEFORE attempting to program this device. If the LCD Animator Setup Utility works and the program you have been writing does not, the problem is software related.

QBasic Example Software Problems

We have provided many programming examples in QBasic. These programs should run without problems, with one very notable exception, QBasic MUST BE RUN FROM DOS. Running QBasic under the DOS shell in Windows 95 will NOT work. Our tests have shown that some data scheduled for transmission out the serial port simply gets lost. If you are desperate to run these programs under QBasic within the Windows 95 operating system, put delays of a universal length after EVERY RS-232 transmission. These delays will not be required when running QBasic from DOS.

Last Resort

If you are still having problems with communication to the LCD Animator, a couple of hidden buttons may solve your problem if you are running a particularly fast system (e.g, faster than a Pentium 120). These buttons slow down communication to the LCD Animator. Our tests show that these buttons should NEVER BE REQUIRED. But just in case they are, they can be accessed by maximizing the window of the LCD Animator Setup Utility. These buttons are labeled "GO SLOW" and "GO VERY SLOW." Go Slow tells the program to count to 1000 between every transmission to the display. Go Very Slow tells the program to use 2000 as the count delay. When both are checked, the program will have a count delay of 5000. Please call Ryan Sheldon at (404) 244-2432 if this solves your problem. Again, THESE BUTTONS SHOULD NEVER BE REQUIRED.

Technical Assistance

If your problems persist, please contact Ryan Sheldon at (404) 244-2432. Please be ready to access your computer and the LCD Animator hardware BEFORE calling.

LCD Animator Setup Utility Source Code

The LCD Animator Visual Basic 4 source code is available upon written request at no additional cost. Please include your name, company, address, and phone number with your request. Please write: National Control Devices, P.O. Box 384, Osceola, MO 64776.

We welcome suggestions and recommendations. Please tell us about any errors or omissions you may find in this manual. All recommendations become the property of NCD.

The Networkable LCD Animator is property of National Control Devices.
Copyright © 1998, 1999 All Rights Reserved.

Windows95 is a registered trademark of Microsoft Corporation.
IBM is a registered trademark of International Business Machines.
Macintosh is a registered trademark of Apple Computer.
All Rights Reserved.

For Technical Assistance, Please Contact:
Ryan Sheldon
Phone/FAX: (404) 244-2432
E-mail: ncdryan@aol.com

For a Current Catalog, Please Contact:
National Control Devices
Box 384
Osceola, MO 64776
Fax: (417) 646-8302
E-mail: ncdorders@aol.com

Gray-Scale Imaging

for the
The LCD Animator 128

Users Guide

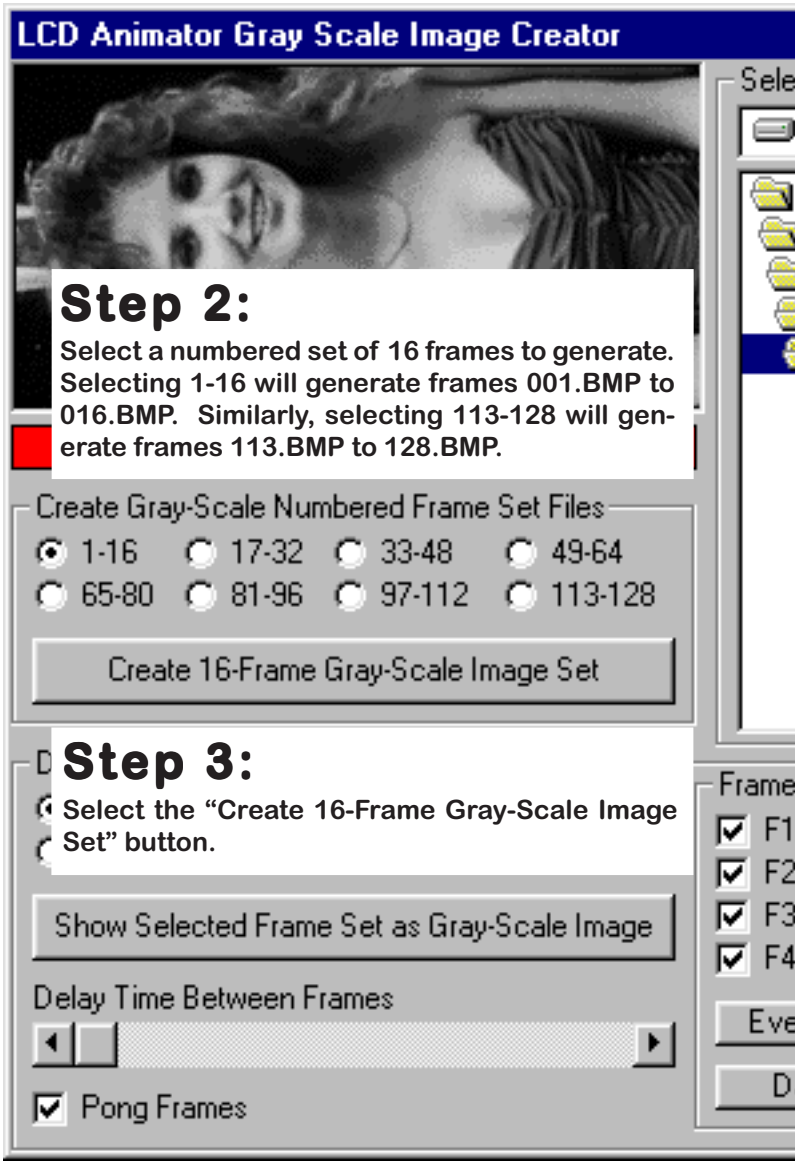
Revision 1.0

By Ryan Sheldon

Copyright © 1998, National Control Devices

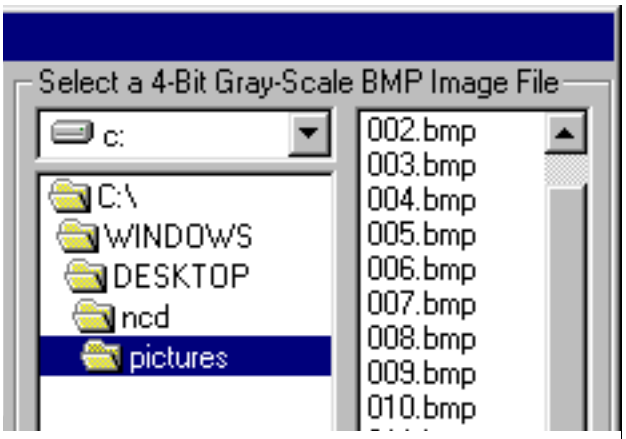
About Gray-Scale Images:

The LCD Animator 128 if capable of displaying up to 8 16-level gray-scale images or 16 8-level gray-scale images. The Gray-Scale Image Creator takes a 16 level gray-scale image and converts it to a set of 16 frames that can be animated at high-speed. When these frames are animated, a gray-scale image is formed. Because of the naturally slow responce time of LCD displays, the image appears solid. To create a gray-scale image, follow the steps outlined in the diagram. Note that this program creates the files and displays the animation, it does NOT load the images into the LCD Animator. Use the LCD Animator Setup Utility and the Batch Send command to send the newly-created frames to the LCD Animator.



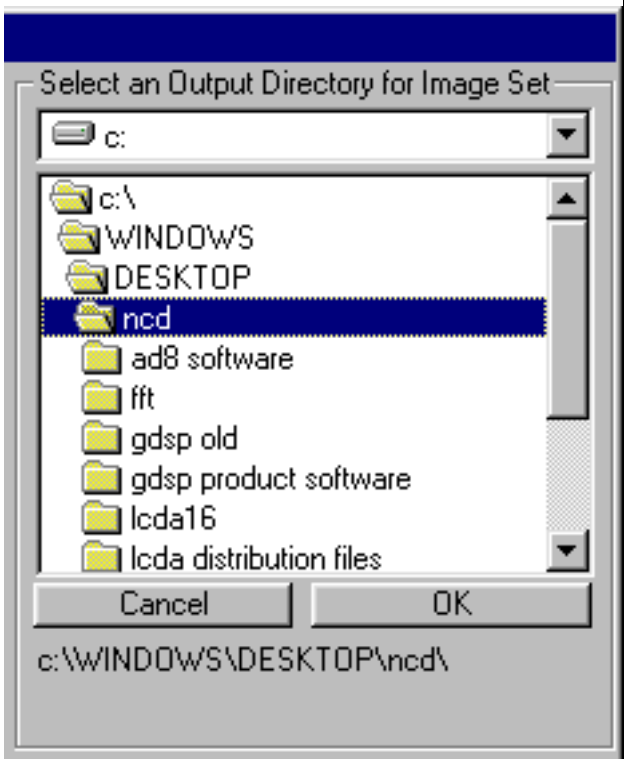
Step 1:

Use the file requestor to select and display a gray-scale image to convert. The picture in the upper left corner of the screen will update to the newly-selected picture.



Step 4:

The following dialog will appear. Select a directory that will hold your 16-frame animation. Then select "OK". Select "Cancel" to select a different gray-scale image.



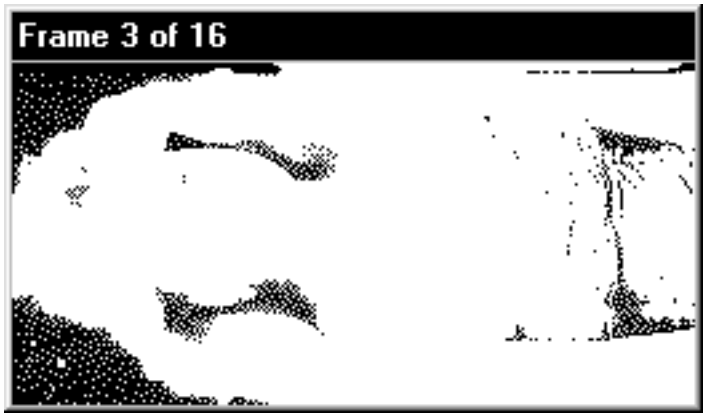
Step 5:

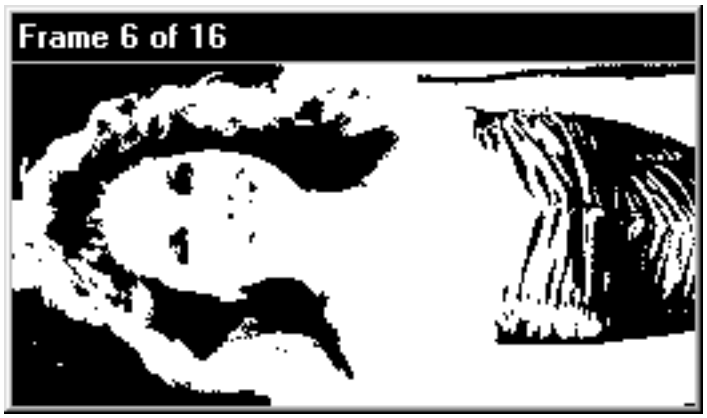
After a few moments. The program will generate a set of 16 frames. The sequence (right) is an example of 3 frames from a 16-frame gray-scale image. After the image set has been generated, you will be returned to the LCD Animator Setup Utility. Use this program to upload your images into the LCD Animator 128. Select the directory that holds your newly-created gray-scale animation and select the "Batch Send" button.

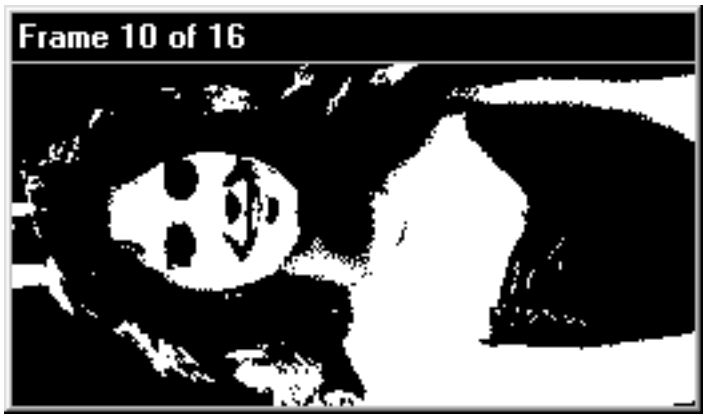
After the images have been uploaded, return to the Gray-Scale

Step 6:

Return to the LCD Animator Gray Scale Image Creator. Now select the range of frames that contain your gray-scale animation, then select the "Show Selected Frame Set as Gray-Scale Image" button to view.







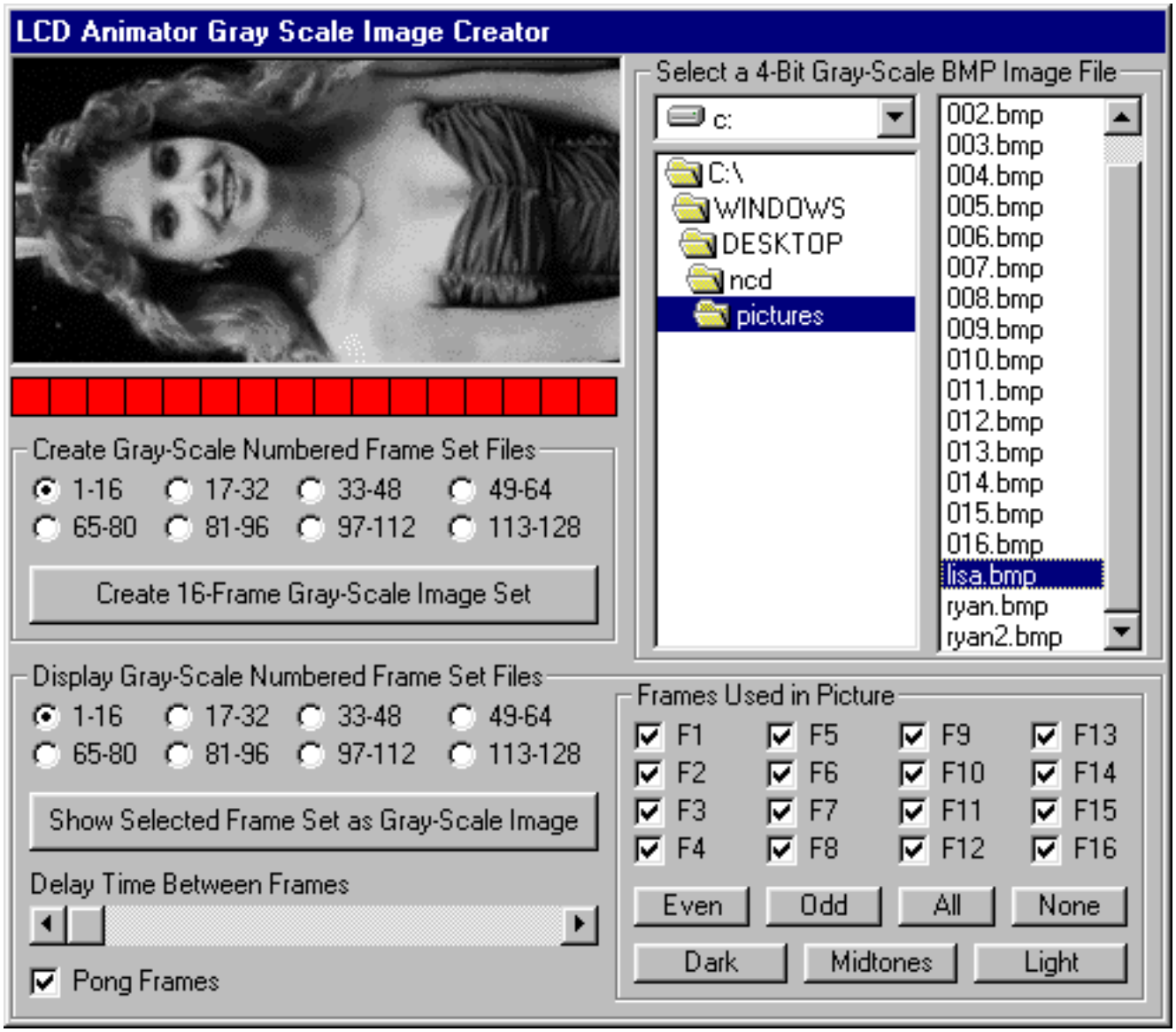
Programming:

The following QBAsic example demonstrates how to display a 16-level gray-scale animation using only a few lines of code.

Once called, this subroutine will never exit. Print #1 is used in the example to send an ASCII character out the serial port of your computer and requires the #1 channel to be opened. Please see the programming examples for the LCD Animator 128 for additional examples on how to open and use the serial port.

For best results, use 19.2K baud with 8 data bits and 2 stop-bits. If you are using 1 stop bit, place a 14 μS selay after the "Optional Delay" remark. A delay will be required at 38.4K and 115.2K baud, but is not necessary if you are using 2 stop bits at 19.2K or less.

```
Procedure Gray (Device, Chapter)
   'Compute the Device Number
   Dev = (Device + 8)*16
   'Change to the chapter that holds
   'the 16-level gray-scale animation
   Print #, Chr$(Dev + 5 + Chapter);
   'Setup a continuous loop
   'There is no escape in this Example
   Do
      'Display frames 0 to 15
      For Gra = 0 to 15
         'Send frame-change Command
         Print #1, Chr$(Dev + 4);
         'Send Frame to change to
         Print #1, Chr$(Dev + Gra);
         'Optional Delay
      Next Gra
   Loop
End Proc
```



Delay Time:

Use the delay slider to adjust the delay between frame changes. This slider also creates some interesting visual effects.

Pong Frames:

Pong Frames stabilizes the gray-scale image by playing frames forward, then backward in a continuous loop. Frame 1 is the lightest frame, and frame 16 the darkest. When frame 1 is played after frame 16, a flicker is produced because of the contrast between these frames. Pong Frames eliminates this flicker.

Frames Used:

Turning off these checks will skip the playback of the cooresponding frame. You may find that a 16-level gray-scale image looks just as good with 8 frames.

Presets:

Even, Odd, All, None, Dark, Midtones, and Light are preset check-mark settings. Dark darkens the image by selecting the dark frames. Even selects only the even frames. Experiment with these settings and optimize your gray-scale image to use as few frames as possible.

We welcome suggestions and recommendations. Please tell us about any errors or omissions you may find in this manual. All recommendations become the property of NCD.

The Networkable LCD Animator is property of National Control Devices.
Copyright © 1998, 1999, All Rights Reserved.

Windows95 is a registered trademark of Microsoft Corporation.

For Technical Assistance, Please Contact:
Ryan Sheldon
Phone/FAX: (404) 244-2432
E-mail: ncdryan@aol.com

For a Current Catalog, Please Contact:
National Control Devices
Box 384
Osceola, MO 64776
Fax: (417) 646-8302
E-mail: ncdorders@aol.com